VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Redukce bezškálových grafů pomocí genetických algoritmů

Scale-free Network Reduction by Genetic Algorithms

2014 Martin Střílka

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Martin Střílka**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Redukce bezeškálových grafů pomocí genetických algoritmů
Scale-free Network Reduction by Genetic Algorithms

Zásady pro vypracování:

Cílem práce je navrhnout a implementovat algoritmus pro redukci bezeškálových grafů pomocí metody genetických algoritmů. Algoritmus bude hledat takový redukovaný graf, který bude mít obdobné vlastnosti (např. distribuci stupňů vrcholů) jako původní neredukovaný graf.

Diplomová práce bude tedy zahrnovat:
1. Přehled vlastností charaketrizujících bezeškálové grafy.
2. Přehled metod používaných pro redukci rozsáhlých grafů.
3. Vytvoření nebo nalezení testovacích dat (rozsáhlého bezeškálového grafu).
4. Návrh, implementaci a ověření genetického algoritmu pro redukci bezeškálových grafů.

Seznam doporučené odborné literatury:

Sampling from Large Graphs by J. Leskovec, C. Faloutsos. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2006

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Pavel Krömer, Ph.D.**

Datum zadání: 01.09.2013

Datum odevzdání: 07.05.2014

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.*

V Ostravě 7. května 2014 .....

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 7. května 2014 .....

V prvé řadě chci poděkovat své rodině za trpělivost a poskytnutou podporu v průběhu celého studia. Dále chci poděkovat všem svým přátelům za to, že mi vždy pomohli odreagovat se od každodenních starostí. Chtěl bych také poděkovat Ing. Pavlu Krömerovi, Ph.D. za poskytnuté rady.

Abstrakt

Diplomová práce popisuje návrh a implementaci genetického algoritmu, který je schopen redukovat bezškálové grafy. Bezškálové grafy se vyskytují v celé řadě oborů jako jsou například sociologie, biologie nebo informatika. Vyznačují se především mocninným rozdělením konektivity uzlů a existencí několika málo vysoce propojených center. Pro analýzu sítí existuje mnoho algoritmů počítajících hodnotné metriky (centrality, nejkratší cesty, atd.), ale některé z nich se stávají problematické v případě rozsáhlých grafů. Provádění detailních simulacích některých internetových protokolů na rozsáhlém grafu je časově velice náročné. Pro tyto aplikace je vhodné graf redukovat. Redukční metoda, kterou se zde zabývám je založena na teorii genetických algoritmů. Redukovaný graf by měl mít obdobné vlastnosti jako původní neredukovaný graf, především tvar distribuce stupňů vrcholů.

Klíčová slova: bezškálový graf, genetický algoritmus, redukce grafu

Abstract

This master's thesis describes design and implementation of Genetic Algorithm, which is able to create a representative sample from Scale-free network. Scale-free networks can be found in many fields e.g. sociology, biology or computer science. Main characteristic properties of Scale-free networks are that degree distribution follows power-laws and in network exists few highly connected centers. There are many known algorithms to compute interesting measures e.g. centrality, shortest paths etc. but some of them are impractical for large networks. Also in many applications it is needed to run expensive algorithms e.g. simulations of internet routing protocols. This is the reason why sampling from graph is essential. The method which is described in this thesis is based on theory of Genetic Algorithms. Reduced network should have similar properties as the original network, for example shape of degree distribution.

Keywords: Scale-free network, Genetic Algorithm, network reduction

Seznam použitých zkratek a symbolů

BA	–	Barabási-Albert model
CR	–	Crossover Ratio
DNA	–	Deoxyribonukleová kyselina
FF	–	Forest Fire model
GA	–	Genetický algoritmus
MGA	–	Messy genetický algoritmus
POP	–	Population
RE	–	Random Edge
RNE	–	Random-Node Edge
SNAP	–	Stanford Netowrk Analysis Platform
WS	–	Watts-Strogatz model
XML	–	Extensible Markup Language

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

Při analýzách rozsáhlých grafů mohou nastat problémy s výpočetní náročností algoritmů počítajících standardní metriky. Ve spoustě aplikací je třeba provádět náročné simulace na sítích jako třeba simulování směrovacích protokolů, P2P „gossip" protokolů nebo šíření dopadů virálního marketingu. Například při studiích internetových směrovacích protokolů je třeba provádět detailní simulace Border Gateway protokolu, nebo toku v síti [3]. Ovšem simulace v sítích s počtem uzlů přesahující jednotky tisíc mohou být časově velice náročné. Další z oblastí, kde velikost grafu činí problémy je vizualizace sítí [4]. Ve všech těchto případech je vhodné graf redukovat tak, aby byly zachovány jeho charakteristické vlastnosti a dále při analýze pracovat s redukovaným vzorkem.

Tato práce se nezabývá redukcí obecných grafů, ale pouze těch bezškálových. Ty jsou zajímavé z toho důvodu, že bezškálové vlastnosti se objevují ve spoustě reálných sítích. Řadí se mezi ně třeba sociální sítě, letištní sítě nebo třeba topologická síť internetu. Společnou vlastností těchto sítí je mocninné rozdělení stupňů vrcholů, proto je důležité, aby při redukci byl zachován tvar této distribuce.

Existuje mnoho algoritmů, které jsou schopny redukovat velikost grafů. Obsahem této práce je popis genetického algoritmu, který je schopen sestavit takový redukovaný graf S takový, že jeho vlastnosti jsou škálovatelné na vlastnosti původního grafu G. V kapitole 2.4 představím ty vlastnosti grafu, na jejichž základě lze určit míru podobnosti grafů.

V první kapitole se zmiňuji o bezškálových grafech a jejich charakteristických vlastnostech jako jsou třeba výskyty komunit, center, mocninné vlastnosti a růst. Spolu s nimi představuji vybrané modely sítí a ještě uvádím devět distribucí (vlastností) pro určení míry podobnosti dvou grafů. V další kapitole se zabývám současnými metodami redukce grafů a dvěma odlišnými cíli redukce. Čtvrtá kapitola pojednává obecně o genetických algoritmech, jsou zde popisovány například různé varianty genetických operátorů. Následující kapitola je rozdělena na dvě části, první popisuje návrh genetického algoritmu pro redukci sítí a druhá pojednává o jeho implementaci. Hlavním cílem experimentální části je provedení a vyhodnocení experimentů nad vybranými datovými kolekcemi pomocí implementovaného algoritmu.

2 Bezškálový graf

Za zakladatele teorie grafů je považován švýcarský fyzik a matematik Leonhard Euler, který roku 1736 v Královci řešil úlohu jak najít okružní cestu městem, která vede přes sedm mostů, přičemž každý most musí být při cestě navštíven právě jednou. Euler nahradil každou z oblastí souše uzlem a každý most hranou, dostal tak graf se čtyřmi vrcholy a sedmi hranami. Pak dokázal, že neexistuje žádná cesta, která by procházela každou hranou právě jednou. Tím položil základy teorie grafů, která je dnes základem našich úvah o sítích [1].

Graf (neorientovaný) je definován jako uspořádaná dvojice $G = (V, H)$, kde V je neprázdná množina vrcholů a H je množina hran, neboli neuspořádaných dvojic u, v takových že $u \in V$ a $v \in V$. Stupeň vrcholu je v tomto případě počet hran incidentních s daným vrcholem. Orientovaný graf je také definován jako uspořádaná dvojice $G = (V, H)$, kde V je opět neprázdná konečná množina vrcholů, ale H je množina orientovaných hran, neboli množina uspořádaných dvojic u, v takových, že $u \in V$ a $v \in V$. Říkáme, že orientovaná hrana h je výstupní hranou vrcholu v, když $h = (v, x)$ pro libovolné $x \in V$. Orientovaná hrana h je nazývána vstupní hranou vrcholu v, pokud platí $h = (y, v)$ pro libovolné $y \in V$ [9]. V případě orientovaného grafu je rozlišováno mezi vstupním a výstupním stupňem vrcholu. Vstupní stupeň označuje počet vstupních hran vrcholu. Výstupní stupeň je dán jako počet výstupních hran vrcholu.

Pojem bezškálová síť je poměrně nový, vznikl někdy na přelomu tisíciletí na univerzitě v Notre Dame pod vedením fyzika Alberta-László Barabásiho. Bezškálový jev znamená, že při změně měřítka zůstanou zachovány vlastnosti systému. Příkladem takového jevu může být třeba fraktál. Bezškálový graf má mocninnou distribuci stupňů vrcholů. Tyto sítě se vyskytují ve odvětvích jako sociologie, biologie nebo počítačové vědy. Postupně zde uvedu nejdůležitější vlastnosti bezškálových sítí.

Jedni z prvních, kteří se pokoušeli modelovat reálné sítě pomocí grafů, byli matematici Erdőse a Rényi. Do té doby se teorie grafů zabývala převážně pravidelnými grafy, kde mají všechny vrcholy stejný stupeň jako třeba mřížka, jenže v reálných sítích se pravidelnosti nevyskytují. Do sítí tedy zavedli náhodu a jejich model grafu se dnes nazývá Erdőse-Rényi model. Při vytváření takového grafu se hrany umisťují zcela náhodně. Rozdělení stupňů vrcholů odpovídá Poissonovu rozdělení, které je charakteristické tím, že má výrazné maximum. Většina uzlů má průměrný stupeň a po stranách maxima rozdělení rychle klesá a v důsledku toho jsou odchylky od průměru vzácné. Vztaženo k příkladu lidské společnosti čítající sedm miliard jedinců, by znamenalo, že zhruba každý z nás má průměrný počet přátel. Pravděpodobnost výskytu někoho kdo má výrazně menší nebo větší počet přátel je exponenciálně malá. Jak objasním později, příliš to neodpovídá skutečnosti [1].

2.1 Vlastnosti bezškálových grafů

2.1.1 Malý svět

Roku 1967 Stanley Milgram, profesor psychologie na Harvardu, pořádal experiment, ve kterém chtěl zjistit „vzdálenost“ mezi dvěma libovolnými lidmi v USA. Náhodně vy-

braní lidé v Kansasu nebo Nebrasce měli odeslat dopis některému z jeho přátel v Bostonu. Předpokládal, že adresát a odesílatel se navzájem neznají. Pravidla byla taková, že dopis je možné odeslat pouze člověku, se kterým se odesílatel osobně zná. Bylo odesláno celkem 160 dopisů, ze kterých se mu vrátilo 42, některé měly pouze dva mezičlánky, jiné jich vyžadovaly více než deset. Nicméně průměrný počet prostředníků byl šest. To dalo vzniknout myšlence, že v celé síti miliard lidí je průměrná vzdálenost mezi dvěma libovolně vybranými pouze šest, tedy jsme všichni šest kroků od sebe a tvoříme tzv. Malý svět [10]. Odtud pochází pojem šest stupňů odloučení. Pozdější experimenty na Univerzitě v Notre Dame na části webu čítající 300 000 uzlů ukázaly, že v daném vzorku je průměrná vzdálenost devatenáct kroků. I ostatní experimenty na dané téma prokázaly, že malá světy jsou běžné ve spoustě sítí, jako například druhy v potravním řetězci jsou dva kroky od sebe. Nebo molekuly v buňce jsou od sebe vzdáleny třemi chemickými vazbami. Lze říci, že malé světy jsou typickou vlastností reálných sítí [1].

Průměrná vzdálenost je definována jako průměrná délka nejkratších neorientovaných cest mezi všemi dvojicemi vrcholů. Mějme graf G s n vrcholy, nejkratší vzdálenost mezi vrcholy v_1 a v_2 označíme $d(v_1, v_2)$, pokud neexistuje neorientovaná cesta z v_1 do v_2 nebo pokud platí, že $v_1 = v_2$ pak $d(v_1, v_2) = 0$. Průměrnou vzdálenost v grafu dostaneme jako $L_G = \frac{1}{n(n-1)} \sum_{ij}^{n} d(v_i, v_j)$ [11].

Průměrem grafu označujeme velikost největší cesty z množiny nejkratších cest mezi dvěma libovolnými vrcholy. Efektivní průměr grafu je hodnota rovna délce nejkratší cesty, která je delší než 90 % všech nejkratších cest. V průběhu vývoje sítí bylo pozorováno, že efektivní průměr se s časem zmenšuje [5].

2.1.2 Komunity

Mark Granovetter ve svém velice vlivném sociologickém článku „Síla slabých pout“ [12] přišel s představou společnosti odlišnou od náhodného světa. Dle jeho modelu jsou lidé pospojováni silnými vazbami do shluků, kde každý zná každého (struktura tvoří kompletní graf), a všichni z daného shluku mají několik slabých vazeb na osoby z jiných shluků. Tyto slabé vazby zamezují tomu, aby se společnost rozpadla na spousty izolovaných komunit. Slabé vazby v naší společnosti hrají důležitou roli, například hledáme-li si zaměstnání, často se stává, že naši blízcí přátelé nám nemohou pomoci. Pohybují se ve stejných okruzích přátel a vědí tedy zhruba to co my. Abychom získali novou informaci, musíme použít právě slabé vazby, tedy poradit se s našimi známými, ti se pohybují v jiných kruzích a mohou mít jiné informace. Návrh takové struktury sítě se ovšem neshoduje s Erdőse-Renýim modelem náhodné sítě. V té je totiž pravděpodobnost, že moji přátelé jsou navzájem také přátelé velice nízká [1].

Duncan J. Watts a Steven Strogatz zavedli pojem shlukovací koeficient [13], který je definován pro uzel jako poměr existujících vazeb mezi jeho sousedy k jejich maximálnímu možnému počtu, tedy tak, aby tvořili kompletní graf. Říká, jak dobře jsou vaši přátelé provázání. Aby byla dokázána myšlenka komunit, bylo shlukování měřeno na grafu spolupráce mezi vědci (citační sítě). Kde uzly grafu jsou autoři a pokud společně publikovali nějaký článek, tak jsou spojeni hranou. Byl zde zjištěn vysoký shlukovací

koeficient. I v dalších sítích jako například v elektrické rozvodné síti na západě USA, kde uzly jsou generátory a transformátory a hrany tvoří vedení vysokého napětí, byla zjištěna stejná skutečnost. Ovšem Erdőse-Renyiho model náhodného grafu má nízký koeficient shlukování. Aby Watts se Strogatzem mohli modelovat sítě s vysokým stupněm shlukování, přišli s novým modelem, dnes nazývaným Watts-Strogatz model. Při generování grafu jsou uzly nejprve uspořádány do kruhu tak, že každý uzel je spojen se svými prvními a druhými nejbližšími sousedy, graf má vlastnosti mřížky. Pro zachování vlastností malého světa je ještě přidáno několik hran náhodně, aby zkrátili cestu mezi vzdálenými uzly, a tedy snížily průměrnou vzdálenost. Tento model dokázal sloučit vlastnosti malého světa s tvorbou komunit, ovšem ani on se příliš neslučuje s reálnými sítěmi, protože vylučuje vznik center [1].

Existují dva způsoby měření shlukovacího koeficientu C. První je definován jako poměr trojnásobného počtu všech trojúhelníků, tedy cyklů délky tři (a) k počtu všech trojic v síti propojených dvěma hranami (b), pak $C^{(1)} = \frac{3a}{b}$. Druhý způsob výpočtu shlukovacího koeficientu je dán jako poměr počtu trojúhelníků jejichž součástí je *i-tý* vrchol (a_i) k počtu trojic se středem ve vrcholu i (b_i), $C_i = \frac{a_i}{b_i}$. Pak shlukovací koeficient pro celou síť je $C^{(2)} = \frac{1}{n} \sum C_i$, kde n je počet vrcholů [14].

2.1.3 Centra

Informace v této kapitole opět pocházejí z [1]. Existuje test, který navrhl reportér Malcolm Gladwell ve své knize „Bod zlomu“ [15], který vám řekne, jak moc společenští jste. Předloží vám seznam 248 jmen vypsaných z telefonního seznamu Manhattenu a když znáte někoho se jménem ze seznamu, připíšete si bod, což ovšem funguje pouze pro osoby amerického původu. Gladewll dal tento test náhodně vybrané skupině sta vysokoškolsky vzdělaných lidí, průměrně účastníci nasbírali 39 bodů. Všiml si, že rozpětí výsledků bylo nečekaně velké od 9 do 118. Přičemž skupina lidí byla velmi homogenní, tvořili ji lidé podobného věku, vzdělání a příjmu. Podobné výsledky se opakovaly i v jiných sociálních skupinách. Došel tedy k závěru: „V každém místě, kam nás život zanese,... jsou tu a tam nemnozí lidé se skutečně výjimečnou dovedností, jak si získat přátele a známé. To jsou prostředníci“. Prostředníci v naší společnosti plní důležitou roli, vytvářejí trendy a módní vlny, jsou pojivem společnosti tím, že spojují skupiny různých kultur, vzdělání, zájmů atd. Gladwell si myslel, že se jedná o něco typicky lidského, ovšem podobné vlastnosti vykazují i jiné reálné sítě.

Při experimentu na Univerzitě v Notre Dame na části webu čítající 203 miliónů uzlů bylo zjištěno, že na celých 90 % stránek míří méně než deset odkazů, zatímco tři stránky mají více než milión vstupních odkazů. Náhodný web by tyto anomálie zcela vylučoval a všechny uzly by si byly velice podobné, je tedy jasné, že tato síť se neřídí náhodným rozdělením vstupních vrcholů.

Další sociální síti, na které je možné studovat její vlastnosti je síť herců, kde uzly jsou tvořeny právě herci a vazba mezi nimi vzniká v případě, že spolu hráli v nějakém snímku. S touto sítí souvisí jeden experiment z 90. let nazvaný Baconovo orákulum [16], kdy skupinka nadšenců považovala herce Kevina Bacona za střed herecké sítě dostupné na IMDB.com. Jedná se o obdobu Erdősova čísla. Byla měřena průměrná vzdálenost od

libovolného herce ke všem ostatním. Ukázalo se, že Bacon není ve středu Hollywoodské sítě (měl průměrnou vzdálenost 2,79), ale na prvním místě byl s hodnotou 2,53 Rod Steinger. Nicméně toto číslo není moc dobrým ukazatelem významnosti uzlu, jelikož rozdíly na předních pozicích jsou velice malé. Jedná se o tzv. „Closeness centralitu“ [14]. Podstatnější zjištění ovšem bylo to, že u významnosti uzlu nezáleží až tak na jeho počtu vazeb jako na tom kam ty vazby vedou. Ústřední pozici měli herci, kteří byli současně zapojeni do mnoha velkých shluků, tedy za svou kariéru vystřídali mnoho filmových žánrů.

Na tomto místě bych ještě měl zmínit Erdősovo číslo – matematik Erdőse byl ve své vědecké činnosti výjimečně plodný a publikoval na 1500 článků s 507 spoluautory. Erdősovo číslo říká, jakou vzdálenost má autor v síti vědecké spolupráce od Erdőse. Erdőse má číslo nula, každý kdo s ním přímo spolupracoval, má číslo jedna. Ukazuje se, že většina matematiků má Erdősovo číslo malé mezi dvěma až pěti. Ale i fyzikové nebo ekonomové jej mají nízké. To dokazuje, že vědecká síť je hustě propojena, tvoří malý svět a Erdőse je možné považovat za jedno z významných center. Spolupracující vědci se navzájem znají, je tedy možné tuto síť považovat za zmenšený model naší sociální sítě, jejich vlastnosti by si měly být podobné.

Centra se objevují také v sítích jako je síť molekul v buňce, kde molekuly jsou spojeny chemickými vazbami. Dále topologická síť internetu spojujících počítače po celém světě, má centra. V telefonní firmě AT&T přišli na to, že podstatná část všech příchozích a odchozích hovorů připadá na malé procento telefonních čísel. Ty nejčastěji představují tzv. call centra.

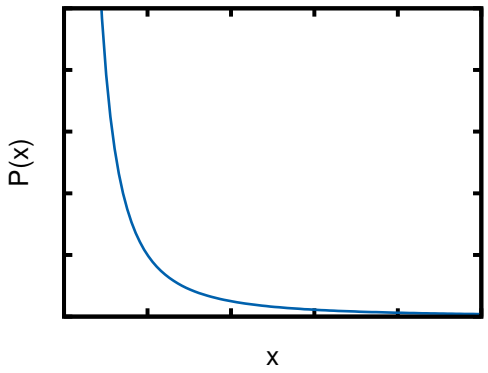
Jak je vidět centra jsou důležitým prvkem reálných sítí, formují strukturu všech sítí, ve kterých se vyskytnou, dělají z nich malé světy. Budují mosty mezi kterýmikoli dvěma místy v systému, z toho vyplývá, že průměrná vzdálenost mezi dvěma uzly v naší sociální síti je sice zhruba šest, ale vzdálenost kohokoli od nějakého prostředníka je často jeden nebo dva kroky. Ovšem Erdőse-Rýnyiho a Watts-Strogatsovy modely s centry nepočítají, proto jsou pro modelování síti nevhodné.

2.1.4 Mocninné vlastnosti

Při experimentech na Univerzitě v Notre Dame na vzorku webu čítajících cirka 300 000 uzlů, zjistili, že histogram konektivity uzlů neodpovídá Poissonově rozdělení, které se řídí Gaussovou (zvonovitou) křivkou, s výrazným maximem. Jejich histogram ukázal, že rozdělení počtu odkazů se řídí mocninným zákonem. V případě Gaussovy křivky by měly všechny uzly podobný průměrný stupeň a jakékoliv odchylky od průměru by byly vzácné, jelikož zvonovitá křivka po stranách klesá s exponenciální rychlostí. To by znamenalo, že všechny dokumenty jsou zhruba stejně propojené a existence center je vyloučena. Graf webu ovšem obsahoval většinu uzlů s nízkou konektivitou a několik center s neobyčejně hodně vazbami. Mocninné zákony vznik takových center umožňují, křivka totiž klesá mnohem pomaleji [1].

Spolu s mocninnými zákony zmíním i Paretovo pravidlo 80/20. Italský ekonom Pareto si všiml, že v určitých oblastech je možné pozorovat zvláštní úkaz. Například 80 % půdy je ve vlastnictví 20 % obyvatel, 80 % zisku produkuje 20 % zaměstnanců, 80 % peněz na

výplaty dostává 20 % nejbohatších lidí atd. Tyto jevy se řídí mocninným zákonem. Ovšem většina veličin se v přírodě řídí Poissonovým rozdělením, například výška obyvatelstva, inteligenční kvocient, rychlost molekul v plynu atd. Kdyby se například výška obyvatelstva řídila mocninným zákonem, většina lidí by byla nízkého vzrůstu, ale nikoho by nepřekvapilo, že sem tam potká tří kilometrového obra. To je významná vlastnost mocninných zákonů, tedy že vedle sebe existuje spousta malých jevů s několika málo velmi velkými. Poměr mezi největším a nejmenším jevem v mocninném rozdělení je mnohonásobně vyšší než v náhodném světě. Třeba poměr výšky nejvyššího člověka (272 cm) k nejnižšímu (74 cm) je zhruba 3,7. Rozdělení počtu obyvatel v amerických městech odpovídá mocninnému rozdělení. Největší je New York s 8,3 milióny obyvateli a nejmenší Duffield ve Virginii s 50 obyvateli, poměr největší k nejmenšímu je zhruba 153 000 [14, 1].

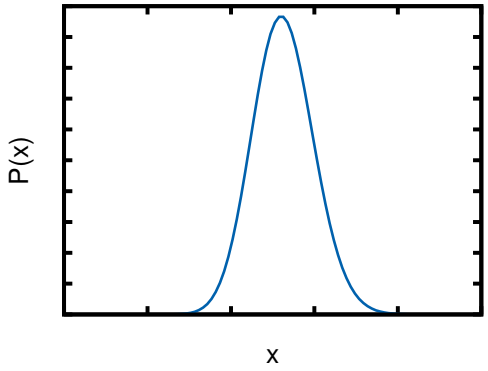


Obrázek 1: Mocninné rozdělení

Každý mocninný zákon je charakterizován exponentem, v případě grafů je nazýván exponentem konektivity, jelikož popisuje rozdělení konektivity uzlů. Funkce $P(k)$ nám říká kolik je webových dokumentů s právě k vazbami. Řídí se vzorcem $P(k) = k^{-\alpha}$, kde α je exponent konektivity. Tento exponent nám například říká, kolikrát méně je oblíbených stránek oproti těm oblíbenějším. Pokud mocninnou křivku vyneseme do grafu s logaritmickými osami, dostaneme klesající přímku a exponent konektivity určuje její sklon. Při testech na vzorku webu, výzkumníci v Notre Dame zjistili, že exponent rozdělení vstupních odkazů se pohybuje kolem 2,1 a exponent konektivity výstupních odkazů byl přibližně 2,5 [1].

Řada komplexních sítí se řídí podobnými zákony jako web, například mocninné zákony byly objeveny i v síti Hollywoodských herců. Kde počet herců s k vazbami klesá podle mocninného zákona. V síti molekul buňky byly tyto zákony taktéž pozorovány. Zjistilo se, že počet molekul reagujících s k jinými molekulami klesá podle mocninného zákona. V ostatních bezškálových sítích se exponent α pohybuje v rozmezí $2 \leq \alpha \leq 3$. Často se také stává, že mocninné vlastnosti reálných sítí se objeví až od určitého stupně uzlu, například v citační síti se mocninné vlastnosti projevují až od vstupního stupně uzlu většího než 100, tedy $x_{min} = 100$ [14].

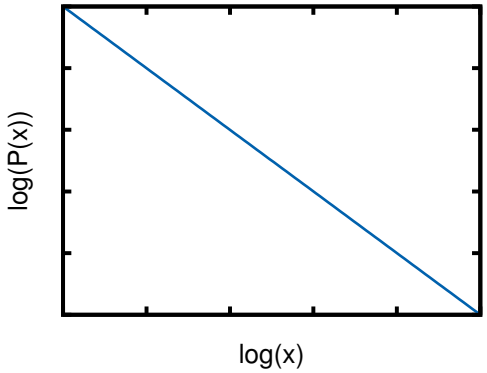
Náhodné sítě tím, že mají výrazné maximum v rozdělení konektivity, tak umožňují existenci něčeho jako průměrného uzlu, tedy lze je charakterizovat typickou hodnotou konektivity uzlů neboli škálou. U sítí s mocninným rozdělením postrádáme maximum, místo toho je zde jen hierarchie uzlů, která sahá od několika center až k množství malinkých uzlů. V této stupnici neexistuje žádný průměrný uzel, který bychom mohli prohlásit za typický uzel sítě. Není zde žádná charakteristická škála, jedná se tedy o bezškálové sítě (v anglickém znění scale-free). Otázkou je, jaký zákon nebo jev stojí za touto vlastností bezškálových sítí? Mocninné zákony se zřídka kdy vyskytují v systémech řízených náhodou. Fyzici zjistili, že často jsou signálem přechodu od neuspořádanosti k řádu, ale v těchto sítích mocninné zákony jsou znamením samo-organizace [1].



Obrázek 2: Poissonovo rozdělení

Poissonovo rozdělení bývá označováno jako rozdělení řídkých jevů, jelikož se dle něj řídí jevy s nízkou četností. Požívá se pro aproximaci binomického rozdělení v případě velkého počtu pokusů. Určuje pravděpodobnost výskytu jevu pro danou průměrnou hodnotu $\lambda = \frac{k}{n}$, kde k je počet událostí, které nastaly za počet jednotek n. Pravděpodobnostní funkce pro náhodnou veličinu X je definována jako $P(X = x) = \frac{\lambda^x}{x!}e^{-\lambda}$ pro $\lambda > 0$. Jak lze vidět na obrázku 2, toto rozdělení je charakteristické svým maximem, většina událostí má tedy průměrnou hodnotu [9].

Mocninná závislost je polynomiální funkce jedné proměnné x s exponentem k. Vypadá následovně $f(x) = ax^k + o(x^k)$, kde a, k jsou konstanty a o je asymptoticky menší funkce. Pro mocninnou funkci platí $f(cx) \propto f(x)$, kde c je konstanta a znamená, že změnou argumentu konstantním poměrem se změní pouze její měřítko, ne tvar. Jinými slovy mocninný zákon vypadá stejně nezávisle na měřítku, jakým se na něj díváme. Mocninný zákon se obvykle znázorňuje jako $\log f(x) = \log b + \alpha \log x$, kde α je „měřítkový exponent“ a představuje sklon závislosti, lze pozorovat nezávislost násobící konstanty b na tvaru funkce f a exponentu α. V logaritmickém měřítku má mocninná křivka tvar přímky. V případě bezškálových sítí se používá tvar $p(x) \approx x^{-\alpha}$, typická hodnota exponentu se pohybuje v rozmezí $2 \leq \alpha \leq 3$ [14].



Obrázek 3: Mocninné rozdělení v logaritmickém měřítku

2.1.5 Růst a preferenční připojování

Náhodný model předpokládá, že všechny uzly sítě máme k dispozici od začátku, jejich počet v průběhu existence sítě neroste ani neklesá, je tedy statická. Dalším předpokladem je, že všechny uzly si jsou navzájem rovnocenné. Nedokážeme je navzájem odlišit a náhodně je spojujeme mezi sebou. V případě komplexních sítí se tyto dva předpoklady ukazují jako nepravdivé. Uvedu na příkladu webu.

V době svého vzniku obsahoval web pouze jediný dokument, později začali nejprve zaměstnanci univerzit přidávat své vlastní webové stránky a síť začala růst. A roste pořád, dnes obsahuje miliardy dokumentů a každý z nich vznikal jeden po druhém a postupně se připojovali k síti. Modelovat rostoucí síť je jednoduché, začínáme s malým počtem uzlů a postupně přidáváme další, tak že se připojí k náhodně vybranému uzlu. Ty, které jsou starší, měly více času na to posbírat více vazeb, nejmenší stupeň bude mít posledně připojený uzel. Nicméně rozdělení stupňů vrcholů, tedy vlastnost, která odlišuje náhodné sítě od bezškálových, klesá příliš rychle. Jinými slovy, pouze růst nestačí proto, aby se síť stala bezškálovou [1].

Je nutné se zamyslet nad tím, jak si my, uživatelé webu, vybíráme, kterou stránku navštívíme. Například v případě, že hledáme portál pro zpravodajství a zadáme do Googlu výraz „zprávy“, dostaneme přibližně 114 000 000 dokumentů. Dle náhodného modelu bychom si z nich vybírali zcela náhodně. Místo toho si ale vybereme ten, který je nám více sympatický. Pokud bychom dostali dva zpravodajské servery, z nichž na jeden míří dvakrát více odkazů než na druhý, většina lidí si vybere ten s větší konektivitou. Řídíme se principem preferenčního připojování. To platí nejen pro web, ale i mezi herci je pravidlem, že čím více herec natočil filmů, tím větší má šanci být obsazen i do dalších filmů. Předchozí model růstu sítě rozšíříme o preferenční připojování, ve smyslu, že nový uzel se bude připojovat ke stávajícím úměrně jejich konektivitě. Tedy uzly s vyšším stupněm mají vyšší pravděpodobnost, že se k nim připojí další uzly, starší vrcholy jsou ve výhodě, poněvadž stihly posbírat více vazeb než nově příchozí. Neformálně řečeno platí „bohatí bohatnou, chudí chudnou“. Tento jednoduchý model, pojmenován po svých tvůrcích je nazýván Barabási-Albert model, je schopen generovat bezškálové sítě zahrnující několik

málo center, tedy nejstarších uzlů, a spoustu vrcholů s malým stupněm. Pouze dvě jednoduchá pravidla, růst a preferenční připojování, stačí na to, aby se v systému objevily mocninné zákony [1].

Náhodné a Watts-Strogatzovy modely se zaměřovaly pouze na strukturu sítě, BA model se zabývá také mechanismem jejího vzniku a dokázal, že struktura a způsob vzniku sítě se od sebe nedají odloučit. Bylo pozorováno, že v průběhu vývoje reálných sítí se jednak zmenšuje efektivní průměr a také se zvětšuje průměrný stupeň vrcholu, tedy sítě se stávají „hustšími“ [5].

2.2 Shrnutí vlastností bezškálových sítí

Na tomto místě shrnu výše zmíněné charakteristické vlastnosti bezškálových sítí.

- nízká průměrná vzdálenost mezi vrcholy (Malý svět)
- vysoký koeficient shlukování (komunity)
- vysoce propojená centra, která „drží sítě pohromadě“
- mocninné rozdělení distribuce stupňů
- neexistuje průměrný uzel
- sítě jsou dynamické - rostou, nově příchozí uzly se řídí preferenčním připojováním
- univerzální v tom smyslu, že nezávisí na specifických vlastnostech domény
- neměnnost vůči měřítku, mocninný zákon vypadá stejně, nezávisle na měřítku, s jakým jej pozorujeme, přímka v logaritmických osách má pořád stejný sklon [14]

Už jsem ukázal, že bezškálové sítě se vyskytují v celé řadě oblastí. Jsou pro nás zajímavé tím, že jejich studiem můžeme objasnit některé běžně se vyskytující jevy. Zde jsou ty nejznámější oblasti.

- sociální sítě a sítě spolupráce (herecká síť, síť vědecké spolupráce)
- počítačové sítě zahrnující topologickou síť internetu i síť dokumentů World Wide Web
- některé finanční sítě jako třeba síť mezibankovních platebních styků
- sémantické sítě jako třeba lexikální síť
- síť interakcí proteinů a molekul v buňce
- letištní síť

Výskyt bezškálových sítí tímto výčtem nekončí, avšak bezškálová podstata spousty dalších sítí je stále diskutována spolu s vývojem datových analýz.

2.3 Modely sítí

2.3.1 Erdőse-Rényi náhodný model

Existují dvě varianty tohoto modelu, první $G(n, p)$, kde n reprezentuje počet vrcholů a p je pravděpodobnost s jakou jsou libovolné dva vrcholy spojeny hranou. V $G(n, m)$ modelu je mezi n vrcholů vloženo m hran s uniformní pravděpodobností. Obě varianty jsou schopny vytvářet jak orientované tak neorientované grafy. Více v [17].

Shrnutí vlastností náhodného modelu

- malá průměrná vzdálenost
- nízký shlukovací koeficient
- absence center
- distribuce stupňů vrcholů odpovídá Poissonově rozdělení
- statický

2.3.2 Watts-Strogatzův model

Model je dán předpisem $WS(n, k\ , p)$, modeluje náhodný svět tím, že na počátku je n vrcholů uspořádaných do kruhu a každý z nich má vazby na k svých nejbližších sousedů. Vlastnosti malého světa jsou přidány tím, že každá hrana je přepojena s pravděpodobností p k jinému náhodně zvolenému vrcholu. Tento model dává vzniknout neorientovaným grafům. Více v [17].

Shrnutí vlastností Watts-Strogatzova modelu

- malá průměrná vzdálenost
- vysoký shlukovací koeficient
- absence center
- distribuce stupňů vrcholů odpovídá Poissonově rozdělení
- statický

2.3.3 Barabási-Albert model

Grafy vytvořené dle tohoto modelu mají vlastnosti bezškálových sítích, proto jej zde popíšu detailněji.

2.3.3.1 Postup generování Na začátku je graf s m_0 vrcholy, nové uzly se připojují k m existujícím vrcholům, musí tedy platit $m \leq m_0$. V každém kroku se připojí právě jeden nový uzel a pravděpodobnost, že se připojí ke stávajícímu uzlu i s k_i vazbami je rovna $p_i = \frac{k_i}{\sum_{j=0} k_j}$.

Existuje také rozšířený BA model, který v jedné iteraci zahrnuje tři operace. S pravděpodobností p přidá m nových hran, s pravděpodobností q přepojí m hran a s pravděpodobností $1–p–q$ přidá nový vrchol. Tato varianta je schopna produkovat realističtější sítě [14].

2.3.3.2 Distribuce stupňů Distribuce stupňů odpovídá mocninnému rozdělení $P(k) \sim k^{-3}$, graf roste lineárně a mocninné zákony platí pro všechny stupně, což u reálných sítí není moc časté, jelikož u reálných sítí mocninné zákony jsou často kombinovány s exponenciálním začátkem nebo koncem. Také dává do souvislosti stáří a stupeň vrcholu, což ne vždy odpovídá reálným sítím.

Pravděpodobnost, že bude v síti existovat m vrcholů s nejvyšším stupněm k je $\binom{n}{k} p_k^m (1 - P_k)^{n-m}$, kde P_k je kumulativní pravděpodobnost. Potom pravděpodobnost, že nejvyšší stupeň v síti bude k je $h_k = \sum_{m=1}^{n} \binom{n}{k} p_k^m (1-P_k)^{n-m} = (p_k + 1 - P_k)^n - (1-P_k)^n$ a očekávaná hodnota nejvyššího stupně je $k_{max} = \sum_k k h_k$ [18, 9].

2.3.3.3 Průměrná vzdálenost Průměrná vzdálenost je nízká, nižší než v náhodných sítích. V průběhu času se s rostoucí sítí průměrná vzdálenost zvyšuje. Je dána vztahem $L \sim \frac{\ln(N)}{\ln(\ln(N))}$ [18, 9].

2.3.3.4 Shlukovací koeficient Shlukovací koeficient vrcholu i stupně k je dán jako $C_i = \frac{2n_i}{|k_i(k_i-1)|}$, kde n_i je počet hran vedoucí k k_i sousedům vrcholu i. Bylo zjištěno, že shlukovací koeficient sítě je závislý na její velikosti $C \sim N^{-0,75}$ [18, 9].

2.3.3.5 Varianta modelu se zdatností Barabási a Albertová navrhli variantu, která odstraňuje souvislost mezi stářím vrcholu a jeho stupněm. Do modelu zahrnuli tzv. zdatnost, preferenční připojování se pak neřídí jen stupněm vrcholu ale i jeho zdatností. Připojení nového uzlu ke stávajícímu, který má k vazeb a zdatnost η, je dána jako pravděpodobnost $p = \frac{k\eta}{\sum_{i=0}^{n} k_i \eta_i}$. V reálných sítích se také vyskytují uzly, které přišly pozdě, ale přesto se ještě byly schopny stát významnými centry. Typickým příkladem je Google, který je dnes nejrozšířenějším vyhledávačem, ale před ním už existovaly třeba Yahoo nebo AltaVista. Toto rozšíření umožňuje vznik dalšího jevu, kdy jeden uzel k sobě připojí téměř všechny vazby v síti. V typické bezškálové síti platí, bohatí bohatnou a chudí chudnou, největší centrum je v hierarchii vrcholů následováno druhým o něco menším a tak dále až k nejmenšímu vrcholu. Jenže existují i sítě, kde vítěz bere vše (Boseho-Einsteinova kondenzace), kde největší centrum přetváří topologii sítě na hvězdu, jejíž je středem.

Příkladem takové sítě, by mohla být síť operačních systémů, kde operační systémy bojují o uživatele tedy o hrany. Pokaždé když si nějaký uživatel nainstaluje Windows, přibude uzlu Windows vazba. V tomto případě má Windows 86 % všech vazeb, následován Mac OS s 5 %. Bezškálové vlastnosti se ovšem z takových sítích vytrácejí [1].

2.3.4 Další modely bezškálových sítí

V navrženém algoritmu používám pro generování počáteční populace ještě další dva modely, pro se zde o nich jen ve zkratce zmíním.

2.3.5 Copying model

Model popisuje bezškálové sítě, které se neřídí jen preferenčním připojováním, například web. Ale snaží se napodobit chování tvůrců webových stránek, kdy bylo pozorováno, že lidé často kopírují stránky svých přátel, když vytvářejí své vlastní. Tímto mechanismem vznikají v grafu komunity [6].

Copying model je dán jako $C(n, k, p)$, kde n je velikost grafu, k je výstupní stupeň uzlu a p je pravděpodobnost s jakou se nový vrchol připojí k náhodnému vrcholu. Proces generování začíná s malým grafem, ke kterému se v každém kroku připojí jeden vrchol. Nový vrchol si náhodně vybere jeden stávající uzel jako *prototyp*. Připojuje se k výstupními hranami, kde pro každou hranu se s pravděpodobností p vybere vstupní vrchol náhodně a s pravděpodobností $1 - p$ se kopíruje hrana z prototypu. Tato druhá část mechanismu zajišťuje vznik vysoce propojených uzlů. Exponent konektivity (vstupního stupně) je větší než dva a řídí se podle vzorce $\alpha = \frac{2-p}{1-p}$.

Tohle je příklad lineárně rostoucí sítě, ale web roste exponenciálně, proto byly vyvinuty i varianty kdy modelovaná síť roste exponenciálně, v každém kroku se připojí množina uzlů, jejíž velikost je rovna konstantnímu zlomku velikosti celého grafu.

2.3.6 Forest Fire model

Tento model byl navržen při studiu vlastností při vývoji grafu. Je schopen vytvářet grafy, kde i distribuce výstupních stupňů vrcholů má tvar mocninné křivky, ačkoli ne tak strmé jako distribuce vstupních stupňů [5].

Vrcholy se připojují po jednom v každém kroku. Každý nový vrchol si vybere „gravitační centrum“ v určité části sítě a pravděpodobnost připojení k libovolnému uzlu klesá s jeho vzdáleností od zvoleného „gravitačního centra“. Některé nové uzly se k síti připojí velmi vysokým počtem výstupních hran, to zajišťuje strmější distribuci výstupních stupňů. S přibývajícím počtem uzlů klesá efektivní průměr a zvyšuje se hustota (poměr hran a uzlů).

Model je dán jako $F(n, p, f)$, kde n je výsledný počet vrcholů, p představuje pravděpodobnost dopředného hoření a r je pravděpodobnost zpětného hoření. Připojení nového vrcholu zahrnuje následující kroky

1. Nově příchozí vrchol v si náhodně vybere ambasadora w, ke kterému se připojí.

2. Dále jsou vygenerována náhodná čísla x, y s geometrickou pravděpodobností a středem $p/(1-p)$ a $rp/(1-rp)$ respektive. Vrchol v pak vybere x výstupních hran a y vstupních hran vrcholu w incidentních s ještě nenavštívenými vrcholy, které si označí jako $w_1, w_2, ..., w_{x+y}$

3. Vrchol v se připojí ke všem $w_1, w_2, ..., w_{x+y}$ a zároveň pro všechny tyto vrcholy se provede krok číslo dvě rekurzivně. Jednou navštívené vrcholy nemohou být navštíveny znovu a algoritmus běží tak dlouho, dokud „oheň vyhasne“.

2.4 Vlastnosti grafů pro určení podobnosti

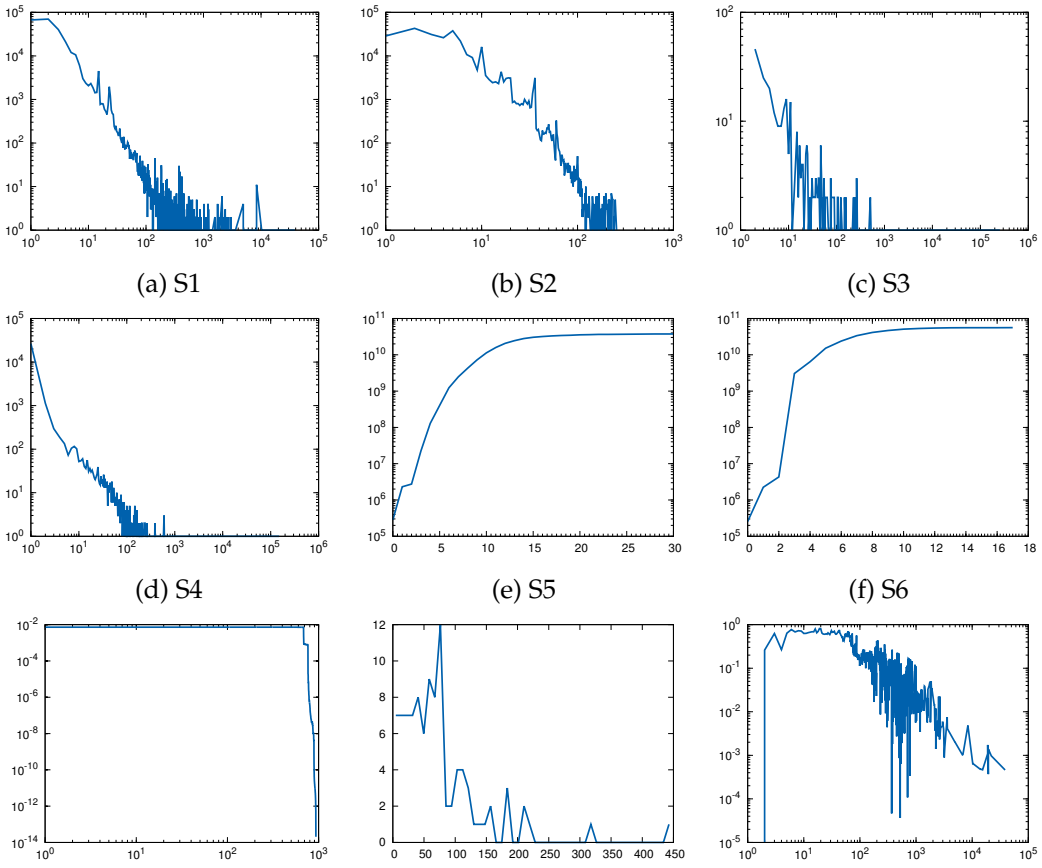
Při redukování sítě je potřeba zjistit jak kvalitní je odvozený vzorek, tedy jak jsou jeho vlastnosti podobné původnímu grafu, případně škálovatelné na vlastnosti původního grafu. Následujících devět vlastností (distribucí) přebírám z [3], kde byly navrženy pro určení podobnosti orientovaných grafů při jejich redukci.

- **S1: Distribuce vstupních stupňů:** histogram všech vstupních stupňů vrcholů v grafu. Každý stupeň d je na ose x, v ose y jsou vyneseny počty uzlů s daným stupněm d.
- **S2: Distribuce výstupních stupňů:** histogram všech výstupních stupňů vrcholů v grafu.
- **S3: Distribuce velikostí slabě souvislých komponent:** množina uzlů je slabě souvislá, pokud pro libovolné dva uzly u a v existuje neorientovaná cesta z u do v. Jinými slovy komponenta je slabě souvislá pokud tvoří souvislý graf. Tyto komponenty je možné nalézt pomocí algoritmu prohledávání do hloubky [19].
- **S4: Distribuce velikostí silně souvislých komponent:** množina uzlů je silně souvislá, pokud pro libovolné dva uzly u a v existuje orientovaná cesta z u do v a z v do u. Tedy komponenta je silně souvislá pokud každý její vrchol je dosažitelný z jiného jejího libovolného vrcholu po orientované cestě. Tyto komponenty je možné nalézt pomocí algoritmu prohledávání do šířky [19].
- **S5: Hop-Plot:** křivka, která ukazuje, kolik párů vrcholů je od sebe vzdáleno n a méně kroků. Číslo $P(h)$ je počet párů, jejichž vzájemná vzdálenost je menší nebo rovna h. Vrchol může tvořit pár i se sebou samým, také každý pár je započítán dva krát. V ose x jsou hodnoty vzdáleností h a na ose y je příslušná hodnota $P(h)$. Hop-Plot kvantifikuje konektivitu a vzdálenost mezi uzly sítě, studuje komunity s určitou vzdáleností a nezabývá se vzdáleností samotnou [20].
- **S6: Hop-Plot na největší slabě zapojené komponentě**
- **S7: Distribuce prvního levého singulárního vektoru matice sousednosti vzhledem k její hodnosti:** první levý singulární vektor je získán jako první sloupec z unitární matice U při SVD rozkladu matice, kde původní matice M o velikosti $m \times n$ je rozložena na $M = U\Sigma V^*$, kde U je unitární matice $m \times m$, Σ je $m \times n$

diagonální matice s kladnými čísly na diagonále, V^* je transponovaná matice V $n \times n$, opět se jedná o unitární matici. Hodnost matice určuje kolik má lineárně nezávislých řádků. Na ose x se nacházejí hodnosti, na ose y jsou hodnoty vektoru.

- **S8: Distribuce singulárních hodnot matice sousednosti vzhledem k její hodnosti:** singulární hodnoty jsou získané jako hodnoty na diagonále z matice Σ z SVD rozkladu. Spektrální vlastnosti grafu obvykle tvoří distribuci s „dlouhým ocasem“.
- **S9: Distribuce shlukovacího koeficientu:** shlukovací koeficient C_d je definován následovně. Nechť uzel v má k sousedů, pak může mezi nimi existovat maximálně $k(k-1)/2$ hran. Nechť C_v představuje zlomek hran, které skutečně existují, k jejich maximálnímu možnému počtu C_v. Pak C_d je definován jako průměr všech C_v pro všechny vrcholy s daným stupněm d. Na ose x jsou vyneseny stupně vrcholů, v ose y je průměrný shlukovací koeficient C_d pro všechny vrcholy se stupněm d.

Na obrázku 4 jsou vyobrazeny distribuce části webu. Tento vzorek pochází z roku 2002 z univerzitní sítě Stanfordu. Všechny grafy mají v obou osách logaritmické měřítko, kromě S5 a S6, které mají logaritmické měřítko pouze v ose y a S8, která je pouze ve standardním měřítku. Z S1 a S2 distribucí lze pozorovat, že se jedná o bezškálovou síť. Dle distribuce slabě souvislých komponent (S3) je možné vysledovat roztříštěnost tohoto grafu na izolované podgrafy. Distribuce shlukovacího koeficientu ukazuje, že se v síti vyskytuje spousta komunit a jejich shlukovací koeficient se s rostoucí velikostí komunity snižuje.



(a) S1 (b) S2 (c) S3

(d) S4 (e) S5 (f) S6

(g) S7 (h) S8 (i) S9

Obrázek 4: Vlastnosti univerzitního webu na Stanfordově univerzitě

3 Redukce grafů

V případě, že dostaneme rozsáhlý graf s milióny vrcholů, jak můžeme odvodit vzorek s podobnými vlastnostmi? Jak velký může být redukovaný graf a jak lze změřit jeho kvalita? Pro otázku měření úspěchu je třeba brát v úvahu i to zda chceme, aby redukovaný graf S měl podobné („zmenšené“) vlastnosti jako původní graf G? Nebo chceme, aby se graf S podobal takovému stavu grafu G v době, kdy G měl velikost S. Tedy bavíme se o dvou možných cílech redukce *Scale-down* a *Back-in-time*. Protože neexistuje česká terminologie, tak dále budu používat anglickou. Níže popíšu několik vybraných algoritmů pro odvozován redukovaných grafů, které byly představeny v [3], kde byly pro měření úspěchu vybraných algoritmů použity vlastnosti pro porovnávání grafů zmíněné v předchozí kapitole.

V případě *Scale-down* redukce je třeba z grafu G s n vrcholy odvodit graf S s n' vrcholů, přičemž $n' \ll n$. Vytvořený graf S musí mít podobné vlastnosti jako G.

Cílem *Back-in-time* redukce je napodobit stav grafu G v určitém časovém bodě jeho vývoje. Přičemž ale známe jen výslednou podobu G. $G_{n'}$ představuje graf G v době kdy měl n' vrcholů, úkolem je nalézt graf S s n' vrcholy tak, aby měl podobné vlastnosti jako $G_{n'}$. Ovšem informace o stáří vrcholů chybí, pokud bychom ji měli, tak redukce by mohla mít pouze podobu odebrání nových vrcholů.

3.1 Současné metody používané pro redukci rozsáhlých grafů

Redukční algoritmy můžeme rozdělit do tří různých skupin: metody založené na náhodném výběru hran, náhodném výběru vrcholů a techniky simulující náhodné procházky nebo šíření virů. Informace o všech následujících metodách jsem čerpal z [3].

3.1.1 Metody založené na náhodném výběru vrcholů

Nejpřímočařejší metodou je *Random Node*, která s uniformní pravděpodobností z cílového grafu vybere množinu vrcholů, které tvoří redukovaný graf. Bylo zjištěno, že vzorek získaný touto metodou nezachovává mocninné vlastnosti cílového grafu.

Další metody jsou založeny na výběru vrcholů proporčně vůči některé z centralit. Metoda *Random Page Rank Node* vybírá vrcholy s pravděpodobností úměrnou jejich *Page-Ranku*, dobře zachovává tvar distribuce vstupních stupňů a hop-plot. Dalším podobným algoritmem je *Random Degree Node*, u kterého se pravděpodobnost vybrání vrcholu odvíjí od vrcholové *Degree centrality*. Tato metoda je ještě více zaměřena na vrcholy s vysokým stupněm, proto příliš nezachovává mocninné vlastnosti sítě.

3.1.2 Metody založené na náhodném výběru hran

Pracují podobně jako metody založené na náhodném výběru vrcholů. Redukční algoritmus *Random Edge (RE)* vybírá hrany s uniformní pravděpodobností, tak dlouho dokud výsledný graf nenabyl požadované velikosti. S touto metodou se váže několik problémů,

výsledný graf bude velmi řídce propojen, takže průměr bude vysoký a komunitní struktura nebude zachována. *Random Node-Edge (RNE)* je mírnou variací předchozího, nejprve se s uniformní pravděpodobností vybere vrchol, pak se náhodně vybere hrana s ním incidentní. RE mírně zvýhodňuje vrcholy s vysokým stupněm, protože je s nimi incidentních více hran, RNE je zase mírně znevýhodňuje. *Hybridní redukční algoritmus* kombinuje oba předchozí. S pravděpodobností p se provede krok RNE, a s pravděpodobností $1–p$ se provede krok RE (typická hodnota pravděpodobnosti $p = 0,8$). Všechny tyto metody umí dobře zachovat tvar distribuce slabě zapojených komponent, ale v celkovém součtu pro všech devět distribucí podávají nejhorší výsledky ze zde zmíněných redukčních algoritmů.

3.1.3 Metody založené na prohledávání grafu

Společným prvkem těchto algoritmů je to, že se nejprve náhodně s uniformní pravděpodobností vyberou vrchol, ze kterého se následně prozkoumává jeho okolí. Výsledkem těchto postupů je spojitý graf. Všechny algoritmy této kategorie jsou schopny zachovat tvar distribuce shlukovacího koeficientu. Metoda *Random Node Neighbor* vybere zcela náhodně libovolný vrchol spolu se všemi jeho výstupními hranami. Což je vhodné pro opravdu velké grafy jelikož imituje čtení ze souboru, kde graf je reprezentován jako seznam hran. Dobře zachovává tvar distribuce výstupních stupňů, ale tvar distribuce vstupních stupňů nezachovává. Komunitní struktura se v redukovaném vzorku také vytrácí.

Další dva algoritmy jsou založeny na algoritmu náhodné procházky grafem, prvním je redukční technika *Random Walk*. U níž je počáteční vrchol vybrán náhodně s uniformní pravděpodobností, a z něj je simulována náhodná procházka. V každém kroku se s určitou pravděpodobností c (obvykle $c = 0,15$) algoritmus vrátí na počáteční vrchol, odkud začíná novou cestu. Problém nastává ve chvíli, kdy počáteční vrchol leží v malé izolované komponentě, proto je dobrým postupem kontrolovat zda po provedení každého kroku bylo navštíveno dostatečné množství vrcholů. Pokud ne, tak se vybere nový počáteční vrchol a algoritmus se spustí znova. Tato metoda dokáže zachovat tvar distribuce vstupních stupňů, singulárních hodnot i prvního levého singulárního vektoru.

Podobně pracuje i redukční metoda *Random Jump*, která se v každém kroku s pravděpodobností c nevrátí na počáteční vrchol, ale náhodně vybere libovolný jiný vrchol, odkud pokračuje v cestě. Tento postup odstraňuje problém uváznutí algoritmu *Random Walk*.

Další metoda je založena na *Forest Fire* algoritmu, který byl představen v publikaci [5], kde je algoritmus použit pro simulaci vývoje grafu v čase. Tato modifikace náhodně vybere počáteční vrchol a začne „zapalovat“ jeho výstupní hrany a uzly s nimi incidentní. Pokud je vrchol zapálen, tak s určitou pravděpodobností zapálí své výstupní hrany. To se opakuje tak dlouho, dokud nebylo „zapáleno“ dostatečné množství vrcholů. Vstupní argumenty metody jsou dopředná (p_f) a zpětná (p_b) pravděpodobnost zapálení. Tato metoda dokáže dobře zachovat tvar distribuce vstupních stupňů, hop-plot, singulárních hodnot i prvního levého singulárního vektoru.

Pro Scale-down redukci nejlepší výsledky podávají techniky založené na algoritmu náhodné procházky grafem. Pro Back-in-time redukci se nejvíce osvědčily metody simulující „Forest-fire“. Minimální velikost vzorku, kdy tyto dva algoritmy ještě podávaly dostatečně dobré výsledky, byla experimentálně stanovena na 15 % původního grafu.

4 Genetické algoritmy

Tyto optimalizační metody se řadí mezi stochastické algoritmy. Obecně se optimalizační algoritmy dělí na deterministické a stochastické. Deterministický algoritmus je předvídatelný, tedy při stejných vstupních parametrech se bude chovat stejně. Mezi deterministické optimalizační techniky řadíme například gradientní metody, které jsou schopny nalézt optimum v nějakém okolí. Druhou skupinou jsou stochastické algoritmy, které nezaručují, že naleznou optimální řešení v konečném čase, ale jsou schopny nalézt sub optimální řešení v přijatelném čase. Tyto metody využívají tzv. heuristik a určitou roli při vykonávání algoritmu hraje také náhoda, tedy při stejných vstupních parametrech se výsledky algoritmu mohou lišit. Do této skupiny patří evoluční algoritmy, mezi které se řadí také genetické algoritmy.

Práce na genetických algoritmech se datují někdy do šedesátých let dvacátého století, kdy J. H. Holland, americký informatik a psycholog, formuloval ideu algoritmů napodobujících biologickou evoluci nejen pro optimalizační úlohy, ale také pro studium adaptivního chování [8]. I z tohoto důvodu jsou genetické algoritmy blíže biologickému modelu než ostatní evoluční algoritmy. Tomu ovšem předcházely objevy Charlese Darwina, které popsal v knize „O původu druhů prostřednictvím přírodního výběru aneb Záchrana preferovaných ras v existenčním boji“ [21]. Kde tento britský přírodovědec představil evoluční teorii dnes nazývanou jako Darwinova evoluční teorie. Nemůžeme opomenout ani jiného neméně významného vědce a to Johana Gregora Mendela, který je považován za zakladatele genetiky a objevitele základních zákonů dědičnosti. Mendel zkoumal křížení rostlin hrachu, kde sledoval zděděné vlastnosti v potomcích, na základě tohoto výzkumu formulovat tři pravidla, kterým se dnes říká Mendelovy zákony dědičnosti. Na poznatcích těchto dvou badatelů stojí dnešní teorie genetických algoritmů. Většina terminologie byla také převzata z biologie.

Stejně jako v biologii pracují genetické algoritmy s populací jedinců, kde každý jedinec má svůj chromozóm. V přírodě mají jedinci zpravidla více chromozómů, ovšem genetické algoritmy pracují spíše s jedinci s jedním chromozómem. Ale existují i verze algoritmů s diploidními jedinci, tedy majícími dva chromozómy. Roli životního prostředí tady hraje fitness (ohodnocovací) funkce, jejíž hodnota určuje jak dobře je jedinec adaptovaný na dané prostředí. Darwinův přirozený výběr je simulován různými selekčními metodami pro výběr jedinců pro následnou rekombinaci, tedy křížení a mutaci. Po vytvoření nových jedinců (potomků) a po jejich ohodnocení jsou lepší jedinci zachováni do další generace a ti horší jsou smazáni, a pak cyklus začíná na novo. Tato evoluce simuluje prohledávání prostoru řešení [22].

4.1 Základní pojmy

Chromozóm – reprezentuje DNA jedince, skládá se z jednotlivých alel (mohou být například binárně kódovány jako 1 a 0), které tvoří větší celky – geny.

Jedinec (fenotyp) – každý jedinec má sadu parametrů, které se nazývají geny. Jedná se o jedno z řešení daného problému.

Populace – množina potenciálních řešení, tedy jedinců

Generace – je tvořena aktuální populací a její číslo určuje počet předchozích iterací algoritmu

Fitness hodnota (vhodnost) – určuje jak dobrý je jedinec, tedy jak dobře je přizpůsobený na dané prostředí (problém). V textu budu dále používat anglický název fitness.

Fitness funkce – každému jedinci v populaci přiřazuje fitness hodnotu, která určuje kvalitu daného řešení. Je obvykle úzce spjata s daným problémem proto její vývoj může vyžadovat velice dobrou znalost domény problému. Je to právě ona, která má největší vliv na vykonávání evoluce a jejíž extrém hledáme, tak by ji měla být věnována maximální pozornost. Vzhledem k tomu, že tato funkce bude volána velice často, je důležité, aby evaluace jedince, byla relativně rychlá.

Počáteční populace – jedná se o důležitou součást GA, jelikož výrazně ovlivňuje rychlost konvergence, vygenerování počáteční populace se zpravidla provádí náhodně tak, aby rovnoměrně pokryla prohledávaný prostor řešení. S dobrou znalostí domény problému je možné vhodně navrhnout počáteční populaci, tak aby urychlila konvergenci algoritmu ke globálnímu optimu.

Selekce – funkce, která vybírá zpravidla dva jedince pro reprodukci, řídí se Darwinovým přirozeným výběrem

Křížení – hlavní rozlišovací znak genetických algoritmů, na vstupu tohoto operátoru jsou dva rodiče a výstupem je jeden případně dva potomci, kteří kombinují vlastnosti svých rodičů

Mutace – důležitý genetický operátor, díky kterému je možné do populace zanášet nové geny a také pomáhá předejít předčasné konvergenci algoritmu. Mutace je považována za hnací motor evoluce, bez které by nemohly vznikat nové vlastnosti jedinců.

Populační výměna – po vytvoření potomků je důležitým faktorem, který ovlivňuje chování GA, také určení, které jedince zahrnout do nové populace a které naopak odstranit

4.2 Princip činnosti genetického algoritmu

Běh jednoduchého genetického algoritmu je možné charakterizovat následujícím předpisem, kde $P(k)$ reprezentuje populaci jedinců *k-té* generace [7].

```
k = 0
 Inicializace  P(k)
Ohodnocení P(k)
Opakuj
    k = k+1
    Selekce P(k) z˜jedinců v˜P(k−1)
    Rekombinace P(k)
    Ohodnocení P(k)
Dokud není splněna terminální podmínka
```

Výpis 1: Pseudokód genetického algoritmu

V pseudokódu 1 vidíme standardní průběh algoritmu, zde přiblížím jednotlivé příkazy

- Inicializace – generování počáteční populace
- Ohodnocení – každému jedinci je přiřazena fitness hodnota
- Selekce – výběr jedinců pro rekombinaci
- Rekombinace – vstupem jsou dva jedinci (rodič) a výstupem jsou jeden nebo dva potomci, vykonávají se zde operace křížení a mutace
- Terminální podmínka – zpravidla se volí určitý počet generací, po kterých se algoritmus ukončí

4.3 Základní operace

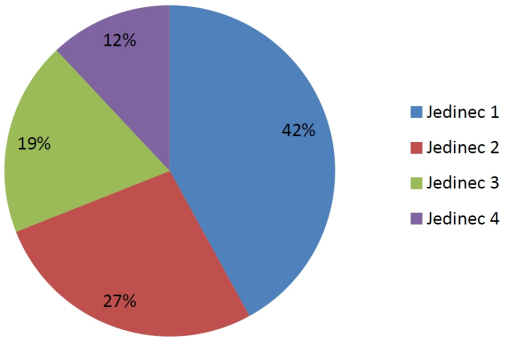
Následující genetické operátory tvoří základ genetických algoritmů, proto je jim věnována celá následující kapitola. Budu zde popisovat pouze základy, tyto informace jsem čerpal z [2], kde se nachází detailnější popisy níže zmíněných operátorů.

4.3.1 Selekce

Tato funkce vybírá rodiče pro reprodukci, pravděpodobnost výběru by měla být proporční k hodnotě fitness každého jedince. Popíšu zde tři nejpoužívanější způsoby vybírání jedinců pro křížení.

4.3.1.1 Proporční selekce Jedná se o metodu představenou v jednoduchém genetické algoritmu nazývaným Hollandovým schématem. Pravděpodobnost výběru *i-tého* jedince je rovna $p_i = \frac{f_i}{\sum_{j=1}^{n} f_j}$ kde n je počet jedinců v populaci, f je fitness hodnota jedince. Někdy se jí také říká ruletová selekce. Na obrázku 5 je možné pozorovat, že na ruletovém kole každý jedinec zabírá takový prostor, jaký odpovídá hodnotě jeho fitness. Tedy lepší jedinci

dostanou více prostoru a ti horší naopak méně. Komplikace mohou nastat v případě, kdy se objevují tzv. super jedinci s příliš vysokou fitness oproti ostatním, potom je jednou z možností upravit fitness funkci, nebo také volba následujícího selekčního modelu.



Obrázek 5: Znázornění proporční selekce

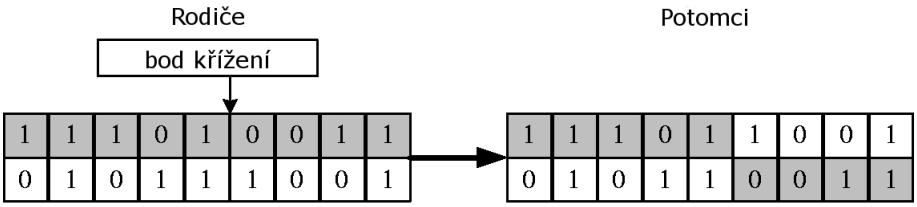
4.3.1.2 Rank selekce Celá populace je vzestupně seřazena podle fitness hodnoty a každému jedinci je přiřazena hodnota (rank) podle pořadí. Parametrem této operace je číslo n udávající poměr rank hodnoty mezi nejlepším a nejhorším jedincem. Jinými slovy nejlepší jedinec má *n-krát* vyšší pravděpodobnost být vybrán pro rekombinaci. Rank hodnota je distribuována mezi jedince vzestupně s konstantním krokem. S hodnotou rank se potom počítá jako s fitness hodnotou v předchozím modelu. Rank selekce sice dokáže eliminovat dominantní super jedince a jiné negativní jevy, ale podstatná nevýhoda je právě v tom, že obaluje fitness hodnotu, což ve stagnující populaci, kdy jsou malé rozdíly mezi fitness hodnotami napříč celou populací, rank selekce uměle zvyšuje selekční tlak, což může mít negativní vliv.

4.3.1.3 Selekce turnajem Existuje více variant tohoto modelu, nejběžnější je *k-turnaj*, kdy je vybráno k jedinců z populace a pro křížení se vezme ten s nejlepší fitness hodnotou. Vhodnou volbou k je možné tuto variantu přizpůsobit na požadovanou úroveň selekčního tlaku.

4.3.2 Křížení

Opět rozlišujeme několik druhů, budu se zabývat těmi, které jsou vhodné pro binární reprezentaci chromozomu. Nejjednodušší forma křížení pracuje s dvěma rodiči, rozřízne jejich chromozom v náhodně zvoleném bodě, zkopíruje je do dvou potomků, kde prohodí odříznuté kousky DNA.

4.3.2.1 Jedno bodové křížení Náhodně je zvoleno jedno místo v chromozomu (alela), kde se provede řez, který je možno pozorovat na obrázku číslo 6. V tomto bodě se prohodí části chromozómu rodičů a zkopírují do jednoho nebo dvou potomků.



Obrázek 6: Znázornění bodového křížení [24]

4.3.2.2 Více bodové křížení Jedná se o přirozené rozšíření jednobodového křížení, v místo jednoho náhodně zvoleného bodu je jich zvoleno n a kousky chromozómů jsou přehozeny mezi těmito n body. Tato varianta dokáže lépe kombinovat dobré kousky DNA, jelikož pracuje na celém chromozomu. Ovšem s rostoucím n začne působit spíše rušivě a dobré vlastnosti se ztratí v šumu. Proto je dobrým postupem zmenšovat n s rostoucím počtem generací, tím je docíleno postupné zvětšování stavebních bloků chromozómu.

4.3.2.3 Uniformní křížení Jsou dání dva rodiče, každý gen v potomkovi je zkopírován z příslušného genu v rodiči. Pro výběr rodiče je pro celý chromozom vygenerována náhodná binární maska, kde hodnota 1 pro *j-tý* gen znamená, že gen j je zkopírován z prvního rodiče, pokud je hodnota rovna 0, tak je *j-tý* gen zkopírován chromozómu druhého rodiče. Tato metoda se ale příliš nepoužívá.

Výběr vhodné metody křížení závisí na řešeném problému. Sekvenční problémy jako hledání nejkratší cesty obvykle vyžadují jiné způsoby křížení, než výše uvedené, jelikož ty by často generovaly neplatná řešení. Je tedy vždy nutné dobře zvážit, jakou metodu křížení použít, nezřídka kdy znalost domény problému lze uplatnit k vývoji vlastní efektivnější metody křížení. Například lze využít znalosti o existenci vhodných bodů, ve kterých je výhodné dělit chromozóm.

4.3.3 Mutace

Mutace je způsob jak zanést do populace nové genetické informace, tedy objevení mírně odlišných částí prohledávaného prostoru. Pro binární reprezentaci problému se mutace realizuje jako přehození hodnoty bitu. Pravděpodobnost mutace v takovém případě by měla být velice nízká, méně než 10 %, jinak hrozí, že se informace ztratí v „šumu“. V případě reprezentace chromozómu jako posloupnosti celých čísel, mutace může mít podobu nahrazení alely náhodným číslem v daném rozsahu. Existují také adaptivní mutační schémata, která adaptují například pravděpodobnost nebo také způsob mutace. Například může mutační schéma vypadat tak, že v počátcích prohledává prostor uniformě a ke konci lokálně poblíž nalezených řešení. Při navrhování mutace je nutné mít na paměti, že výsledek by měl spadat do prostoru validních řešení.

4.3.4 Schémata pro výměnu generací

Po vytvoření potomků současné generace vyvstává otázka které jedince zařadit do další generace. Tato operace v podstatě určuje délku života jedince, má tedy vliv na konvergenci algoritmu. Následující schémata popisují běžné mechanismy výměny generací. Výběr toho vhodného je často záležitostí experimentů.

4.3.4.1 Generační výměna Celá generace je nahrazena jejími potomky. Může se stát, že fitness hodnota nejlepšího jedince se v průběhu evoluce sníží. Což někdy může zabránit předčasné konvergenci, tedy uvíznutí v lokálním optimu.

4.3.4.2 Elitismus Nejlepší jedinec (nebo n nejlepších jedinců) je přeneseno do nové generace. Někdy může vést k předčasné konvergenci. Speciálním a zároveň široce používaným případem je *Golden Gate Model*, kdy $n = 1$. Pokud je ještě na nejlepšího jedince aplikována mutace, mluvíme o tzv. *slabém elitismu*.

4.3.4.3 Delete-n-last Z populace je smazáno n nejhorších jedinců a nahrazeno potomky. Pokud je $n \ll |POP|$ pak mluvíme o tzv. *Steady-State* nahrazovacím schématu.

4.3.4.4 Delete-n V populaci není nahrazeno n nejhorších jedinců jako v předchozím případě, ale n náhodně, s uniformní pravděpodobností, zvolených jedinců. Což na jednu stranu pomáhá zabraňovat uvíznutí v lokálním extrému, ale na druhou stranu snižuje rychlost konvergence.

4.3.4.5 Turnajová výměna Ze staré populace i z potomků se vybere náhodně n jedinců, ze kterých se do nové generace vybere ten nejlepší.

4.3.5 Souhra genetických operátorů

Aby byl umožněn efektivní běh genetického algoritmu, musí být prohledávání prostoru řešení výhodné z hlediska daného problému. Kritickými faktory pro efektivní běh algoritmu je souhra operátorů selekce, křížení a mutace.

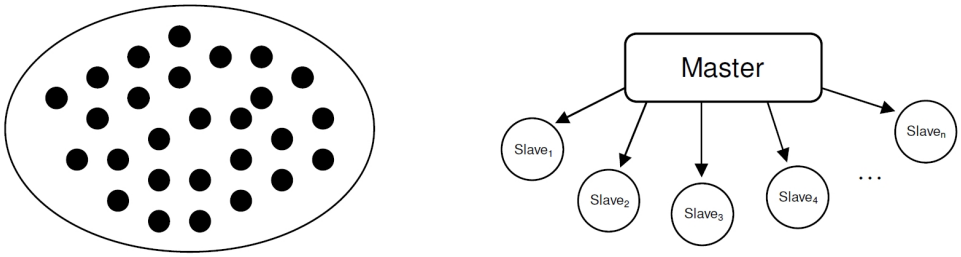
Úkolem křížení je výhodně kombinovat alely vybraných chromozómů z různých částí prohledávaného prostoru. Proto operátor křížení má podobné aspekty jako prohledávání do šířky. Mutace mírně mění určité chromozómy, tím do populace zanáší nové genetické informace a předchází stagnaci. Tím, že mutace mění určité chromozómy jen mírně, je primárně považována za operátor provádějící prohledávání do hloubky. Mutace podporuje aspekty prohledávání do šířky, pokud je křížení schopno nové informace distribuovat do ostatních chromozómů v jiných oblastech prohledávaného prostoru. Tato vlastnost mutace má primární význam pro efektivní fungování algoritmu [2].

4.4 Paralelizace genetického algoritmu

Paralelizace genetického algoritmu je jedním ze způsobů jak urychlit konvergenci, tím že se některé operace rozdělí na menší úkoly a ty se provedou současně na více procesorech. Rozlišujeme tři základní druhy paralelizace, některé zachovávají fungování genetického algoritmu, jiné jej mění. Informace v této kapitole byly čerpány z [2].

4.4.1 Globální paralelizace

Velice podobné jako sekvenční genetické algoritmy, používá se jen jedna populace jedinců, kde každý jedinec má šanci utvořit rodičovský pár s libovolným jiným jedincem. Chování algoritmu zůstává nezměněno, nejčastěji je paralelizované ohodnocování jedinců tedy (účelová) ohodnocovací funkce tak, že úkoly jsou pouze distribuovány mezi více procesy. Procesy se v tomto případě dělí na *master* a *slave*, kdy jediný *master* proces přerozděluje práci mezi podřízené *slave* procesy, které se starají o výpočet účelové funkce na jím přidělení části populace. *Master* proces se stará o běh evoluce (selekce, křížení, mutace).



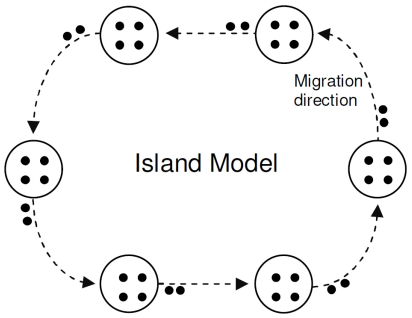
Obrázek 7: Globální paralelní model, převzato z [2]

4.4.2 Ostrovní model (Coarse-grained model)

Také známý jako hrubě dělená paralelizace, kde populace je rozdělena na subpopulace (ostrovy), na kterých evoluce probíhá většinou izolovaně. Výměna jedinců mezi ostrovy se nazývá *migrace* a je jen občasná. Schéma je zobrazeno na obrázku 8. Tento způsob, oproti globální paralelizaci, představuje podstatné změny ve fungování algoritmu, evoluce se tedy chová jinak při než sekvenčním genetickým algoritmem. Hlavní idea za tímto typem paralelizace je ta, že jedinci na každém z relativně izolovaných ostrovů pokryjí jinou část prostoru řešení a při migraci se zkombinují relevantní části DNA. Jedná se o relativně oblíbenou implementaci genetických algoritmů díky tomu, že přirozeně rozšiřuje základní sekvenčního genetického algoritmu.

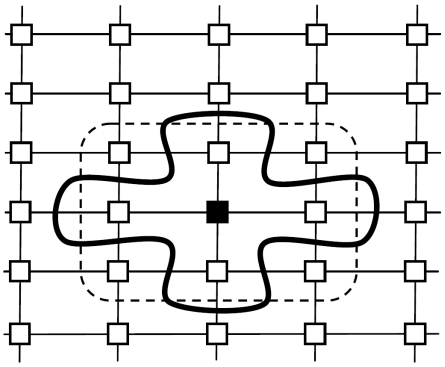
4.4.3 Jemně dělená paralelizace (Fine-grained model)

Tato třída algoritmů má prostorově distribuovanou populaci, kde každý jedinec se může křížit pouze s těmi ze svého blízkého okolí, což je zobrazeno na obrázku 9. Hlavní my-



Obrázek 8: Ostrovní model, převzato z [2]

šlenka spočívá v tom, že jedinci jsou rozprostřeni v prostoru jako molekuly při difuzi. Každému jedinci je přiřazen jeden procesor. Tento způsob paralelizace je vhodný pro masivně paralelní systémy, kde je velké množství slabších procesorů, oproti ostrovnímu modelu paralelizace, pro kterou jsou vhodnější systémy s menším počtem výkonnějších procesorů.



Obrázek 9: Jemně dělená paralelizace, převzato z [2]

4.4.4 Hybridní paralelizace

V poslední době se začínají objevovat také hybridní paralelní genetické algoritmy, nejčastěji se jedná o kombinaci vlastností rozdílných modelů populace. Typickým zástupcem je model, který je na vyšší úrovni hrubě dělený a na nižší úrovni jemně dělený. Nebo také dvouvrstvé modely, které používají hrubě dělenou architekturu v obou vrstvách, kde v nižší vrstvě je vysoká četnost migrace a naopak ve vyšší vrstvě je migrace méně častá.

4.5 Reprezentace problému

Opět platí, že různé problémy lze reprezentovat různě. Při návrhu způsobu reprezentace je nutné myslet na to, že musí umožňovat vykonávání stěžejních operátorů, především křížení a mutace. Dalo by se říci, že úkolem genetického algoritmu je výhodně zkombinovat kusy chromozómu z dobrých jedinců. Tyto kousky DNA se nazývají tzv. stavební bloky, a v průběhu evoluce by se měly zvětšovat. S určitou pravděpodobností jsou tyto stavební bloky rozmístěny už v počáteční populaci, ale jsou rozházeny do všech jedinců. V porovnání s heuristickými metodami, založených na prohledávání okolí jedinců, je tato schopnost kombinovat potenciální řešení výhodou. Proto je nutné daný problém reprezentovat tak, aby tuto schopnost zachoval. Pouze zde zmíním tři nejfrekventovanější způsoby reprezentace. Následující informace jsem opět čerpal z [2].

4.5.1 Binární reprezentace

V první verzi genetického algoritmu byl chromozóm jedince reprezentován řetězcem bitů, které kódovaly třeba celé číslo nebo jejich sekvenci. Jedná se o jednoduchý a stále používaný způsob reprezentace jednice, poněvadž mutaci a křížení lze aplikovat velice jednoduše. Ovšem má to jednu zásadní nevýhodu, kdy rekombinace může způsobit skokovou změnu struktury chromozomu. Čísla lišící se v dekadickém zápise o jedničku se v binárním zápise od sebe mohou odlišovat o více bitů než jeden. To ovšem lze eliminovat použitím tzv. Grayova binárního kódu. Binární reprezentace v Grayově kódu se liší pouze v jednom bitu při změně v dekadickém zápise o jedničku. Díky němu jsou mutace křížení rovnoměrnější a konvergence algoritmu ke globálnímu optimu je také rychlejší.

4.5.2 Reprezentace reálnou hodnotou

Často se volí i reprezentace reálnou hodnotou, kdy má chromozóm podobu vektoru reálných čísel. Ačkoli je pro tyto problémy vhodnější evoluční strategie, bylo vyvinuto několik dobře použitelných metod nepodobných právě evolučním strategiím. Kódování pomocí reálných čísel se odlišuje od většiny typických ostatních úloh tím, že vytváření tzv. stavebních bloků nemá žádný význam. Nicméně pro optimalizační problémy o více dimenzích se ukázalo, že techniky genetických algoritmů jako výběr potomků jsou vysoce účinné i pro tuto oblast.

4.5.3 Reprezentace grafem

Existují také problémy, které jde reprezentovat stromem nebo cestou. K příkladu, pro problém obchodního cestujícího se jedná o způsob reprezentace cestou, kde města jsou seřazena v pořadí, tak jak jdou za sebou. Pořadí měst, která se mají navštívit je dáno n rozměrným vektorem s imaginární hranou mezi prvním a posledním prvkem. Navzdory problémů týkajících se ekvivalence cest (jedna cesta o n vrcholech může být reprezentována n variantami) bylo vyvinuto několik velmi efektivních metod pro rekombinaci.

4.6 Další varianty genetických algoritmů

4.6.1 Adaptivní genetické algoritmy

Jedná se o podskupinu genetických algoritmů, kde parametry jako pravděpodobnost mutace, křížení, velikost populace a další jsou měněny za běhu. Tento postup může zlepšit průběh algoritmu, pokud se podaří správně navrhnout adaptační funkci. Za běhu je vyhodnoceno, zda populace má vhodně nastaveny tyto parametry, pokud ne, tak proběhne jejich adaptace. Idea je taková, že určité nastavení parametrů může být vhodnější pro nalezení obecně dobrého řešení, ale nevhodné pro určení absolutního globálního optima [2].

4.6.2 Messy genetické algoritmy

Hlavním přínosem messy genetických algoritmů je postup výpočtu založený na vývoji stavebních bloků. Stavebním blokem je myšlena spojitá část chromozómu. Myšlenku o stavebních blocích představil David E. Goldberg. Důležitým prvkem hypotézy stavebních bloků je předpoklad o existenci abstraktního mechanismu, který provádí kombinace stavebních bloků. A genetický algoritmus tento mechanizmus nepřímo, ale efektivně implementuje. Messy genetický algoritmus používá chromozómy s proměnnou délkou, geny mají nezávislou pozici, tím že spolu s hodnotou mají také informaci o pozici. Messy algoritmy byly použity v mnoha aplikacích, například na složitých deceptivních funkcích. Messy chromozómy mohou být dvojího druhu, a to podurčené a přeurčené [23].

U přeurčených chromozómů se na stejném místě objevují dvě hodnoty, platí ta dříve použitá a ostatní se ignorují.

V případě podurčeného jedince má chromozóm méně genů než je pozic v chromozómu, tedy v případě binární reprezentace mu chybí určitý počet bitů. Pro výpočet fitness funkce je ovšem zapotřebí celého chromozómu, řeší se to tak, že chybějící bity se zkopírují z tzv. šablony, do které se ukládá zatím nejlepší řešení.

4.6.3 Hybridní genetické algoritmy

Hybridní genetické algoritmy kombinují robustnost genetických algoritmů s rychlostí optimalizačních technik jako jsou například simulované žíhání nebo horolezecký algoritmus. Nejdříve genetický algoritmus nalezne přibližnou oblast s globálním optimem, pak se nasadí lokální prohledávač, který najde extrém. Průběh může vypadat následovně, genetický algoritmus nalezne přibližnou oblast, a jakmile se začne zpomalovat, aktivuje se procedura lokálního prohledávání. Po určitém počtu generací předá procedura lokálního prohledávání své výsledky zpět genetickému algoritmu jako nové chromozómy do populace [2, 23].

5 Návrh a implementace

5.1 Návrh genetického algoritmu

Vstupem genetického algoritmu je bezškálový graf G na n vrcholech a na výstupu je redukovaný graf S na n' vrcholech, přičemž platí $n' \ll n$. Genetický algoritmus pracuje s populací jedinců, tedy množinou řešení, jejichž vlastnosti v průběhu evoluce vhodně upravuj tak, aby se podobaly vlastnostem grafu G. Jedná se tedy o *Scale-down* redukci. *Back-in-time* cílem jsem se nezabýval. Algoritmus také primárně pracuje s orientovanými grafy. Na následujících řádcích popíši, jak jsem navrhl genetický algoritmus tak, aby vlastnosti řešení S konvergovaly k vlastnostem G.

5.1.1 Reprezentace jedince

Pro reprezentaci řešení, tedy grafu, používám maticí sousednosti, což je čtvercová matice $n \times n$, kde n je počet vrcholů grafu S. Hodnota v matici a_{ij} je celé číslo odpovídající počtu hran vedoucích z vrcholu v_i do vrcholu v_j. Tedy $a_{ij} = 1$ pokud v_i je počáteční vrchol a v_j je koncový vrchol hrany e_{ij}. Tato reprezentace je vhodná pro operaci křížení i mutaci, ale grafy s tisíci vrcholy jsou náročné z hlediska alokovaného prostoru operační paměti. Proto po vytvoření jedince křížením a mutací jej převedu na seznamovou reprezentaci, kde každý vrchol je uložen spolu se seznamem vrcholů, se kterými sousedí. Všichni jedinci mají stejný, pevně definovaný počet uzlů, z toho důvodu aby je šlo mezi sebou křížit.

Jinou možností jak reprezentovat jedince, by mohla být ta, že jedinec nebude tvořen grafem, ale bude mít pouze kolekci odkazů na vrcholy původního grafu. Výhoda tohoto přístupu by byla v tom, že by bylo možné vrcholy v redukovaném grafu vztáhnout k vrcholům v původním grafu.

5.1.2 Počáteční populace

Základem rychlé konvergence každého genetického algoritmu je náhodně ale vhodně vygenerovat počáteční populaci jedinců. Mou snahou bylo, aby populace byla co nejrozmanitější. Tedy aby pokrývala různé části prostoru řešení. Jedince generuji různými algoritmy – náhodným modelem ($G(n, m)$), Watts-Strogatz modelem ($WS(n, k, p)$), Barabási-Albert modelem ($BA(n, k)$), Copying modelem ($CP(n, k, p)$) a Forest Fire modelem ($FF(n, p_f, p_b)$). Celkem vytvářím jedince pěti různými způsoby, zastoupení různých jedinců v počáteční populaci je rovnoměrné. Hodnoty parametrů jsou následující:

- $G(n', \frac{e}{n}n')$ – počet hran je volen tak, aby zachovával hustotu původního grafu
- $WS(n', 3, 0.2)$
- $BA(n', 2)$
- $CP(n', 3, 0.2)$
- $FF(n', 0.3, 0.2)$

n – počet vrcholů cílového grafu
e – počet hran cílového grafu
n' – velikost redukovaného grafu

5.1.3 Selekce

Používám selekční model Rank, u kterého lze upravovat hodnotu selekčního tlaku. Hodnota parametru n představuje míru selekčního tlaku a definuje, kolik krát má nejlepší jedinec vyšší pravděpodobnost být vybrán pro rekombinaci oproti tomu nejhoršímu. Platí, že čím nižší hodnota n, tím pomalejší konvergence algoritmu a menší pravděpodobnost uváznutí v lokálním extrému. A naopak čím vyšší hodnota n, tím je konvergence rychlejší, ale je větší nebezpečí, že algoritmus uvázne v lokálním extrému a populace začne stagnovat. Zpravidla volím nízké hodnoty, maximálně velikosti jedné poloviny velikosti populace, tedy $n \leq \frac{|POP|}{2}$, pro populaci čítající 50 jedinců, má tedy nejlepší jedinec 25 krát vyšší šanci výběru. Původně jsem používal ruletovou selekci, ovšem ta mi neumožňovala jednoduše upravovat selekční tlak.

5.1.4 Křížení

Výstupem každého vykonání této operace jsou dva jedinci, zvolil jsem upravenou implementaci vícebodového křížení, které je řízeno parametrem *Corssover Ratio (CR)*. Je vytvářena maska, kde pro každou buňku matice sousednosti se generuje náhodné číslo z intervalu <0, 1) a pokud je menší než *CR*, tak na dané buňce se utvoří bod křížení. Tedy v tomto bodě si potomci navzájem prohodí rodiče, ze kterých si kopírují genetické informace. Tento způsob křížení přináší do klasického n bodového křížení více náhody tím, že pro každé vykonání operace není fixní počet bodů křížení. V průběhu evoluce postupně snižuji *CR* až na předem definovanou minimální hodnotu, abych postupně zvětšoval tzv. stavební bloky chromozómu. Změna parametru křížení je v tomto případě plynulejší než u více bodového křížení. Další výhodou je nezávislost velikosti stavebních bloků na velikosti grafu, což u klasické vícebodové varianty neplatí, jelikož při stejném parametru n s rostoucím grafem roste i velikost stavebních bloků. Počáteční hodnotu volím přibližně 0.25 a finální 0.01, parametr klesá s lineární rychlostí. Experimentoval jsem i s jinými variantami křížení, včetně jednobodové nebo uniformního, ale tato varianta podávala lepší výsledky, tedy algoritmus byl schopen rychleji nalézt kvalitnější řešení.

5.1.5 Mutace

Tento jednoduchý operátor je řízen parametrem p_m. Každá buňka matice sousednosti se při mutaci změní s pravděpodobností p_m. Změnou je myšleno přidání či odebrání hrany, tedy pokud je hodnota buňky $a_{ij} = 1$, tak se hrana odstraní, v opačném případě se hrana přidá.

5.1.6 Evaluace jedinců (Fitness funkce)

Všech devět distribucí jedince porovnávám s distribucemi cílového grafu pomocí dvouvýběrového Kolmogorovova-Smirnova testu, používám z něj pouze hodnotu *D-statistic*. Hodnota D je definována jako $D = max_x\{|F'(x)–F(x)|\}$, kde F a F' jsou dvě kumulativní distribuce a x je hodnota náležící do oboru hodnot obou funkcí, tedy $x \in (H(F) \cup H(F'))$. Dvou výběrový neparametrický Kolmogorovův-Smirnovův test srovnává rozdělení dvou náhodných veličin, v mém případě měřím shodu mezi cílovou distribucí a tou redukovanou. Tím, že se srovnává rozdíl kumulativních četností, tak tato metoda porovnává spíše tvar distribucí, než jejich měřítko. Obě distribuce nejprve převedu na kumulativní, následně hodnoty v ose x převedu do logaritmického měřítka a pak normalizuji tím, že v obou osách převedu všechny hodnoty do intervalu $<0, 1>$. Toho dosáhnu tím, že všechny hodnoty v dané ose vydělím tou největší z nich. Porovnávané empiricky získané distribuce jsou nestejně velké, z pravidla platí $x_r \ll x_c$, kde x_r je maximální hodnota v ose x pro redukovaný graf a x_c je maximální hodnota v ose x cílového grafu. Pro porovnání jejich tvaru je tedy převedení měřítka osy x a normalizace nezbytná. Platí, že čím menší je hodnota D pro danou dvojici distribucí, tím jsou si grafy v dané vlastnosti podobnější. Pro výpočet fitness funkce je možno zvolit libovolnou z devíti vlastností, je ovšem nutné mít na paměti, že čím více jich bude zvoleno, tím pomalejší bude konvergence algoritmu a také stoupá nebezpečí uvíznutí v lokálním minimu.

Jiným přístupem jak realizovat fitness funkci by mohlo být porovnávání grafů založené na porovnání sklonu distribuce stupňů vrcholů (exponent konektivit α) a minimální hodnoty stupně odkud se začínají projevovat mocninné zákony (x_{min}).

5.1.7 Populační výměna

Používám upravený model *Delete-n-last*, kde je nahrazeno alespoň n jedinců v nové generaci potomky. Všichni jedinci, potomci i rodiče, jsou seřazeni podle fitness hodnoty a postupně od nejlepších vkládání do nové generace. Pokud bylo vybráno méně než n potomků, tak dostatečný počet rodičů je nahrazen jinými potomky. V případě, že potomci jsou mimořádně *silní*, se může stát, že do nové generace je vybráno i více než n potomků. Zpravidla volím n rovno jedné třetině velikosti populace.

5.1.8 Paralelizace

Algoritmus je z větší části paralelizován, sekvence křížení, mutace a evaluace mohou běžet simultánně ve více vláknech. Jedná se o globální úroveň paralelizace, to znamená, že algoritmus se bude chovat stejně nezávisle na počtu vláken. Na předem definovaný počet vláken se rozdělí jednotlivé úkoly. Například při mutaci nebo evaluaci potomků dostane každé vlákno určitý počet jedinců na zpracování roven $n = |POP|/t$, kde t je počet vláken. V případě křížení dostane každé vlákno za úkol vyprodukovat počet potomků, který je řízen stejným vztahem jako u mutace nebo evaluace, kolekce rodičů je v tomto případě sdílená mezi všemi vlákny.

5.2 Implementace

Program je implementován v jazyce C++ s kompilátorem VS2013. Používám knihovny třetích stran SNAP a PugiXML. S aplikací se pracuje přes konzoli a její parametry jsou řízeny konfiguračním souborem.

Pro výpočet vlastností grafu je využita knihovna *Stanford Network Analysis Platform (SNAP)* [25] ve verzi 2.1. Jedná se o knihovnu pro analýzu rozsáhlých sítí napsanou v jazyce C++. Umožňuje efektivně pracovat s rozsáhlými grafy, počítat jejich strukturální vlastnosti a v neposlední řadě také generovat sítě podle základních modelů. Tato knihovna vyžaduje pro svůj běh knihovny Gnuplot a Graphviz.

Na počátku běhu algoritmu je načten konfigurační soubor, kterým je algoritmus řízen. Dále je načten cílový graf nebo jsou pouze načteny jeho předem spočtené vlastnosti. Ještě před startem evoluce jsou spočteny distribuce cílového grafu (pokud nebyly načteny). Je možno zvolit, které distribuce se budou počítat do fitness funkce, a které budou sloužit až pro finální zhodnocení výsledku algoritmu. Po vygenerování počáteční populace se spustí samotná evoluce. Algoritmus běží tak dlouho, dokud nebyl dosažen předem definovaný počet generací. Po ukončení evoluce se provede finální ohodnocení nejlepšího nalezeného řešení a výsledky se uloží.

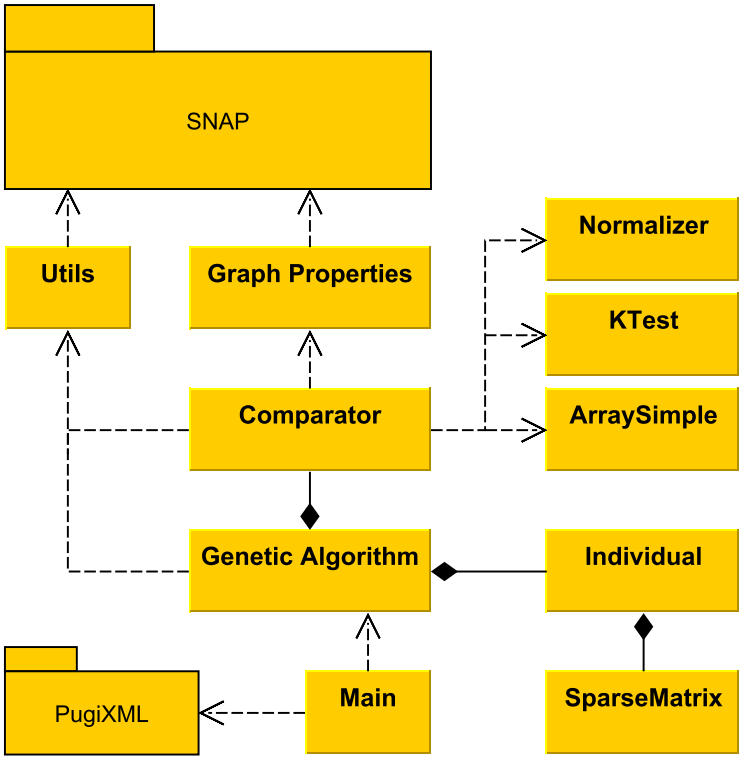
5.2.1 Objektový návrh

Na obrázku 10 je znázorněn třídní diagram programu. Dále se budu věnovat detailnějšímu popisu jednotlivých tříd.

5.2.1.1 GraphProperties Třída, kde je zapouzdřeno počítání vlastností grafu. Obsahuje řadu statických metod S1 až S9, které na vstupu přijímají graf a vracejí požadovanou distribuci v podobě kolekce bodů, každý bod je zde reprezentován párem hodnot x a y.

5.2.1.2 Utils Zde je zapouzdřeno další použití knihovny *SNAP*, například se jedná o statické metody pro ukládání, načítání a generování grafu. Umí také ukládat nebo načítat vypočtené distribuce cílového grafu ze souboru. Každá distribuce je uložena ve zvláštním souboru, jehož struktura vypadá tak, že dvojice hodnot, reprezentující jeden bod křivky, je uložena jako dvě desetinná čísla s plovoucí desetinnou čárkou o velikosti čtyř bytů (datového typu *float*). Každá distribuce je uložena ve zvláštním binárním souboru S1 až S9 končícím příponou *bin*. Spolu s nimi je uložen soubor *info.bin*, kde jsou binárně uložena dvě čísla, první označuje počet vrcholů a druhé počet hran. Dvojice hodnot jsou ukládány za sebou vzestupně podle *x-ové* hodnoty. Společně s třídou *GraphProperties* tvoří třída *Utils* rozhraní mezi genetickým algoritmem a knihovnou *SNAP*.

5.2.1.3 Individual Třída *Individual* reprezentuje jednoduchou strukturu, která v sobě zapouzdřuje informace o jedinci jako jsou jeho fitness hodnota, hodnoty *D-statistic* pro každou z počítaných vlastností, rank, číslo generace, kdy byl vytvořen a samotný graf reprezentovaný seznamem.



Obrázek 10: Diagram tříd

5.2.1.4 SparseMatrix *SparseMatrix* implementuje seznamovou reprezentaci grafu. Tato struktura je realizována maticí a dvěma vektory. V matici indexy řádků odpovídají indexům vrcholů, každý vrchol má ve svém řádku uložené ty indexy vrcholů, které tvoří koncové vrcholy jeho výstupních hran. Ve vektoru *outDeg* je uložen výstupní stupeň každého vrcholu. Jelikož řádky mohou být nestejně dlouhé, tak velikost alokované paměti je zapsaná ve vektoru *arrLength*. Velikosti vektorů a počet řádků matice jsou rovny počtu vrcholů. Tato třída dále zahrnuje metody pro vkládání a odstraňování hrany, které jsou používané při genetické operaci mutace. Pokud při přidávání výstupní hrany k vrcholu je alokovaná velikost řádku rovna výstupnímu stupni vrcholu, tedy nový index už není kam uložit, pak se alokovaný prostor zvětší na dvojnásobek své původní velikosti.

5.2.1.5 ArraySimple *ArraySimple* je jednoduchá struktura, která zapouzdřuje vektor čísel s plovoucí desetinou čárkou, délku daného vektoru a metody pro třídění. Slouží jako datový typ při přenosu informací mezi třídami *Comparator*, *Normalizer* a *KTest*.

5.2.1.6 Normalizer Zde jsou zapouzdřeny operace pro normalizaci distribuce. Na vstupu její hlavní metody *normalize* je distribuce redukovaného grafu a distribuce cílového grafu, výstupem jsou dva vektory hodnot. Nejprve je distribuce převedena na

kumulativní tvar, pak je změněno měřítko v ose x na logaritmické a následně je distribuce převedena do intervalu $<0, 1>$. To je provedeno pro obě distribuce. V posledním kroku jsou vytvořeny vektory hodnot, první je tvořena hodnotami v ose y redukované distribuce. Druhou tvoří y-ové hodnoty z cílové křivky v x-ových bodech té redukované distribuce.

5.2.1.7 KTest Třída *KTest* implementuje Kolmogorovův-Smirnovův test. Používám vlastní implementaci K-S testu, původně jsem hodnotu D počítal pomocí knihovny ROOT, ale ta nefungovala dobře. Pro vstupy do Kolmogorovova-Smirnovova testu jsou brány posloupnosti hodnot z metody *normalize* třídy *Normalizer*.

5.2.1.8 Comparator Třída *Comparator*, jak už název napovídá, porovnává vlastnosti redukovaných grafů vůči vlastnostem cílového grafu. Je zde implementována většina operací pro počítání fitness funkce. Distribuce cílového grafu se buď spočtou před začátkem evoluce, nebo je možné je načíst ze souboru. Lze zvolit, které vlastnosti se budou počítat do fitness funkce a výsledná hodnota fitness každého jedince je sumou všech jeho odchylek D od původního grafu. Vzhledem k tomu, že distribuce jsou normalizované (jejich hodnoty se pohybují v rozsahu $<0, 1>$), tak hodnota D, je také z intervalu $<0, 1>$. Platí, že čím menší D, tím jsou si dané křivky podobnější, tedy jedinec s menší fitness hodnotou je lépe adaptovaný a více se podobá cílovému grafu. Dané distribuce lze porovnávat pouze v případě, že mají více než dva prvky, pokud alespoň jedna z nich toto nesplňuje, tak *Comparator* vrací hodnotu D rovnou jedné. Tato třída řídí tok dat z tříd *GraphProperties* přes *Normalizer* do *KTest*.

Průběh vyhodnocení jedince vypadá následovně - jedinec je předán metodě *compare* objektu *comparator*, která pomocí statických metod třídy *GraphProperties* spočte jeho požadované vlastnosti. Spočtené distribuce jsou předány třídě *Nomalizer*, která je normalizuje a vrátí data připravená pro porovnání. Ty jsou předány třídě KTest, která vrátí spočtenou hodnotu D pro každou z jednotlivých vlastností (distribucí). Jejich suma pak tvoří fitness hodnotu jedince.

5.2.1.9 Genetics Ve třídě *Genetics* je implementován genetický algoritmus. Je zde použit návrhový vzor Singleton pro možnosti paralelizace. Zvolil jsem globální úroveň paralelizace, kdy operace křížení, mutace a evaluace se vykonávají paralelně na n vláknech. Paralelně se také vykonává předgenerování náhodných čísel pro mutaci a křížení. Běh algoritmu je monitorován a zapisován do souboru `log_YYYY-MM-DD_HHMMSS.log`, kde údaje reprezentující čas odpovídají době spuštění algoritmu. Jsou zde zapisovány detailní informace o celé populaci, jako jsou základní informace o každém jedinci, tedy jeho fitness hodnota, hodnoty D pro každou z možných devíti distribucí, aktuální úroveň *CR* atd. Dále se také zaznamenává nejlepší jedinec, ukládá se jeho seznam hran do souboru `best_in_#gen_th_gen.txt`, kde `#gen` označuje číslo generace, kdy byl uložen. Evoluce se spouští metodou *run*. Po jejím doběhnutí se provede finální evaluace nejlepšího nalezeného řešení a jeho seznam hran se uloží do souboru `best_edge_list.txt`.

5.2.1.10 Konfigurační soubor Parametry algoritmu jako jsou velikost populace, počet vrcholů v jedincích, pravděpodobnost mutace a mnoho dalších jsou uloženy v konfiguračním XML souboru *config.xml*, který zde detailněji popíšu. Tento soubor umístěn ve stejném adresáři jako aplikace a je předpokládáno správné nastavení jeho parametrů. Obsahuje následující informace

1. *Crossover* – parametry pro řízení *CR*
   (a) *startRatio* – počáteční hodnota *CR*, je lepší nastavit nějakou větší hodnotu tak, aby stavební bloky byly menší. Typicky 0,25, s níž je dvaceti pěti procentní pravděpodobnost, že se na dané buňce vytvoří bod křížení a průměrná velikost stavebního bloku bude čtyři buňky matice. Parametr je desetinné číslo a může nabývat hodnot z intervalu $<0, 1>$
   (b) *minRatio* – minimální hodnota kam až bude *CR* postupně klesat, typicky 0,01, při této hodnotě je průměrná velikost stavebního bloku sto buněk matice. Parametr má podobu desetinného čísla a leží v intervalu $<0, 1>$
   (c) *speedOfDecreasing* – označuje číslo generace, ve které hodnota *CR* klesne na své minimum. Neboli počet generací, ve kterých se bude parametr *CR* zmenšovat lineární rychlostí, v jedné generaci o hodnotu $\Delta = \frac{startRatio–minRatio}{speedOfDecreasing}$. Aktuální hodnotu *CR* v generaci s pořadovým číslem i je možno spočíst $CR_i = startRatio–\Delta i$, což platí pokud $i \leq speedOfDecreasing$, jinak je $CR_i = minRatio$. Jedná se o celočíselný parametr z intervalu $<0, \infty)$
2. *mutationProbability* – označuje pravděpodobnost, s jakou bude buňka v matici sousednosti zmutována, tedy změněna z nuly na jedna a naopak. Jinými slovy, mutace znamená přidání nebo odebrání jedné hrany z grafu. Typickou hodnotu volím 0,0001, jedná se o desetinné číslo z intervalu $<0, 1>$. Nula znamená, že žádná mutace neproběhne, jedna naopak znamená, že všechny buňky budou zmutovány.
3. *PropertiesForFitness* – parametry, které budou použity pro výpočet fitness funkce. Někdy je vhodnější adaptovat méně než všech devět parametrů grafu. Obecně platí, čím méně je jich zahrnuto, tím rychlejší je konvergence algoritmu k optimálnímu řešení. Jedná se o devět značek (tagů) S1 až S9, jejichž hodnoty mohou nabývat t (*true* – vlastnost je zahrnuta) nebo f (*false* – vlastnost je vyřazen).
4. *PropertiesForFinalEvaluation* – opět se jedná o sadu devíti parametrů, kterými lze nastavit, zdali bude daná vlastnost S1 až S9 zahrnuta do finálního vyhodnocení nejlepšího nalezeného řešení. Pro jejich hodnoty platí totéž co pro *PropertiesForFitness*.
5. *Target* – údaje určující zdroj cílového grafu
   (a) *loadGraph* – parametr, určující zda se bude načítat graf nebo jeho uložené distribuce. Může nabývat pravdivostních hodnot t (pravda) nebo f (nepravda)
   (b) *graphPath* – cesta k souboru s cílovým grafem. Je očekáván textový soubor, který obsahuje seznam hran, každá hrana musí být zapsána na zvláštním řádku

a výstupní vrchol oddělený od vstupního vrcholu znakem tabulátoru nebo středníku

(c) *plotGraphDistrib* – určuje, zda budou distribuce cílového grafu počítané pro finální evaluaci tisknuty Gnuplotem do obrázku. Výstupem jsou tři soubory, první s příponou *plt* reprezentuje skript pro Gnuplot a obsahuje informace o tisknutém grafu jako například jeho velikost. Další soubor, končící příponou *tab*, obsahuje souřadnice bodů distribuce, každý bod je na zvláštním řádku a hodnotay *x y* jsou odděleny znakem tabulátoru. Tento parametr nabývá pravdivostní hodnotu (t/f).

(d) *loadDistrib* – říká, zda se budou načítat uložené distribuce. Hodnota parametru je opět pravda nebo nepravda (t/f) a vylučuje se s parametrem *loadGraph*. Tedy platí, že buď se načte graf, nebo jen jeho předem spočtené distribuce.

(e) *saveDistrib* – opět nabývá hodnot t/f. Pokud má hodnotu pravda, spočtené distribuce se uloží do souborů *S1.bin* až *S9.bin* spolu se souborem *info.bin*, ale pouze v případě, že byl načítán graf, ne pouze jeho distribuce. Pokud je *GA* spouštěn vícekrát pro jeden cílový graf, tak je výhodné distribuce spočíst jednou a uložit do souboru a při každém dalším spuštění načíst.

(f) *distribPath* – je očekávaná cesta k adresáři, kde jsou uloženy soubory s distribucemi, nebo kde se mají spočtené distribuce uložit.

6. *threads* – počet vláken, které aplikace bude při běhu využívat. Jedná se o celé číslo z rozsahu $<1, \infty)$.

7. *sampleSize* – parametr určující velikost (počet vrcholů) redukovaných grafů. Rozsah této celočíselné hodnoty je z intervalu $<1, \infty)$. Příliš nízké hodnoty nemají smysl, jelikož distribuce grafu lze porovnávat, pokud mají více než dva prvky. Doporučená minimální hodnota v tomto případě je 100. Obecně platí čím vyšší tím pomalejší konvergence algoritmu k optimálnímu řešení. Například velikost redukovaného grafu s více jak 1 000 vrcholy je pro algoritmus problematická.

8. *popSize* – velikost populace, minimální počet jedinců v populaci jsou dva (z důvodu křížení), maximální nijak omezen není, pouze hardwarově. Tedy rozsah je v intervalu $<2, \infty)$. Doporučená hodnota je od 50 do 400 jedinců.

9. *generations* – počet iterací evolučního algoritmu. Po poslední z nich proběhne finální ohodnocení nejlepšího jedince. Jedná se o celé číslo z intervalu $<1, \infty)$.

10. *minNumOfChildsInNextGen* – parametr, udávající minimální procentuální zastoupení potomků v nové generaci. Jedná se o desetinné číslo z intervalu $<0, 1>$. Zpravidla volím hodnoty kolem 0,3, což znamená, že v každé další generaci bude alespoň 30 % jedinců novými potomky.

11. *saveBestEveryGen* – celočíselný parametr udávající frekvenci s jakou algoritmus průběžně ukládá nejlepší nalezené řešení. Například hodnota 100 znamená, že po každém vykonání 100 generací se uloží nejlepší nalezené řešení.

12. *maxRank* – udává maximální hodnotu ranku, tedy poměr, kolikrát větší má pravděpodobnost výběru pro rekombinaci nejlepší jedinec oproti nejhoršímu. Jedná se o důležitý parametr ovlivňující selekční tlak a tedy i rychlost konvergence. Čím nižší hodnota tím menší selekční tlak a konvergence je pomalejší. Jedná se o reálné číslo z intervalu $<1, \infty)$.

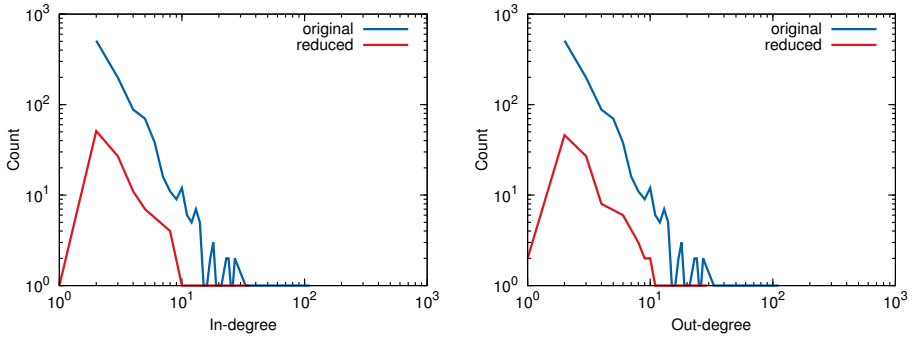
6 Experimenty

Cílem této kapitoly bylo otestovat zda genetický algoritmus při redukci dokáže zachovat bezškálovou topologii sítě tedy hlavně tvar distribuce vrcholů. Pro každou konfiguraci experiment jsem algoritmus spustil alespoň třikrát a vybral nejlepší výsledek.

6.1 Experiment s náhodnými grafy

Cílem tohoto testu bylo ověřit, že v případě, kdy počáteční populace byla vytvořena pouze z náhodných grafů s Poissonovým rozdělením distribuce stupňů, je algoritmus schopen postupnou evolucí tuto distribuci upravit podle cílového bezškálového grafu. Cílový graf byl vygenerován BA modelem s parametry $n = 1000$ vrcholů a počáteční stupeň vrcholu $k = 2$. Počáteční populace grafů byla vygenerována Erdőse-Rényi náhodným modelem s parametry $n = 100$ vrcholů a $m = 399$ hran. Původní graf byl tedy redukován na 10 % své velikosti.

Algoritmus běžel po dobu 2000 generací, fitness hodnota se počítala pouze pro vstupní a výstupní stupně vrcholů. Výstupem byl graf s 347 hranami a hodnoty odchylek jsou shrnuty v tabulce 1. Na obrázku 11 je vyobrazeno srovnání redukovaných a původních distribucí. Je možné si všimnout neshody v počátečních stupních obou redukovaných distribucí s původními. Cílový graf má nejnižší stupeň dva a redukovaný má nejnižší stupeň jedna. Nutno podotknout, že odchylky jsou malé a v obou případech se s postupující evolucí množství vrcholů se stupni jedna postupně snižovalo.



(a) Vstupní stupně (b) Výstupní stupně

Obrázek 11: Srovnání distribucí stupňů redukovaného a původního grafu

Vzorek		S1	S2	S3	S4	S5	S6	S7	S8	S9
vrcholy	hrany									
100	347	0.021	0.054	1.000	1.000	0.151	0.138	0.113	0.320	0.097

Tabulka 1: Výsledky redukce BA modelu

6.2 Vizualizace grafu

Jednou z možných aplikací redukce grafů je vizualizace. Grafy s více jak několika tisíci vrcholy se poměrně špatně zobrazují proto je vhodné graf zredukovat a vizualizovat redukovaný vzorek.

6.2.1 Volební síť Wikepedie (Wikipedia vote network)

Tento experiment byl proveden nad datovou kolekcí volební sítě Wikepedie. Wikipedie je otevřená encyklopedie psaná dobrovolníky z celého světa, někteří z nich mají přístup k některým technickým nástrojům pro údržbu encyklopedie. Tito lidé jsou administrátoři a jsou voleni komunitou ostatních uživatelů. Tato síť představuje volební data od počátku vzniku Wikipedie do ledna roku 2008. Jsou posbírána data 7 115 uživatelů (vrcholy) s celkem 103 689 hlasy (hrany). Orientovaná hrana e_{ij} znamená, že uživatel i volil uživatele j. Většina hlasů v této síti pochází od existujících administrátorů, ostatní jsou od obyčejných uživatelů [26].

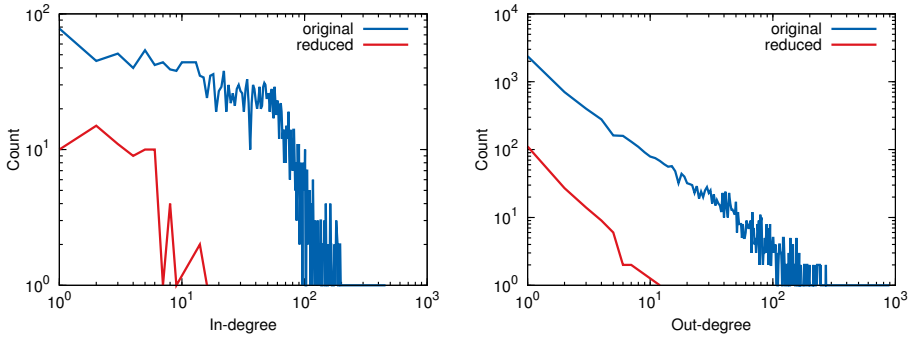
Velikost redukovaného vzorku byla 200 vrcholů, což představuje přibližně 2,8 % velikosti původního grafu. Algoritmus běžel 2000 generací, pro fitness funkci byly zvoleny distribuce S1 a S2. Výstupem byl graf s 311 hranami a hodnotami odchylek od distribucí neredukovaného grafu pro S1 $D = 0,035$ a S2 $D = 0,002$, ostatní odchylky je možné nalézt v tabulce 2 (řádek 2). Na obrázku 12 je zobrazeno srovnání těchto dvou distribucí. Na obrázku 13 je vizualizován redukovaný graf.

Dále jsem pro tuto síť zkoušel stanovit maximální velikost redukovaného vzorku, se kterou dokáže genetický algoritmus pracovat. Výsledky jsou shrnuty v tabulce 2. Pro velikost sítě do 300 vrcholů algoritmus podával dobré výsledky, ale pro síť o velikosti 400 vrcholů už nebyl schopen překonat kvalitu generovaných jedinců počáteční generace, proto jej ani neuvádím ve výsledcích.

Poslední experiment zahrnoval adaptaci všech S1 až S9 vlastností sítě. Zvolil jsem velikost redukovaného vzorku na 150 vrcholů. Výsledný graf po 1 000 generacích měl největší odchylku u S1 (S4 bylo 1 protože původní distribuce měla méně než 3 prvky), ostatní vlastnosti měly malé odchylky. Výsledky jsou shrnuty v tabulce 2 (řádek 4). Adaptovat větší síť bylo problematické (evaluace jedince byla časově náročná a celkově konvergence byla pomalá).

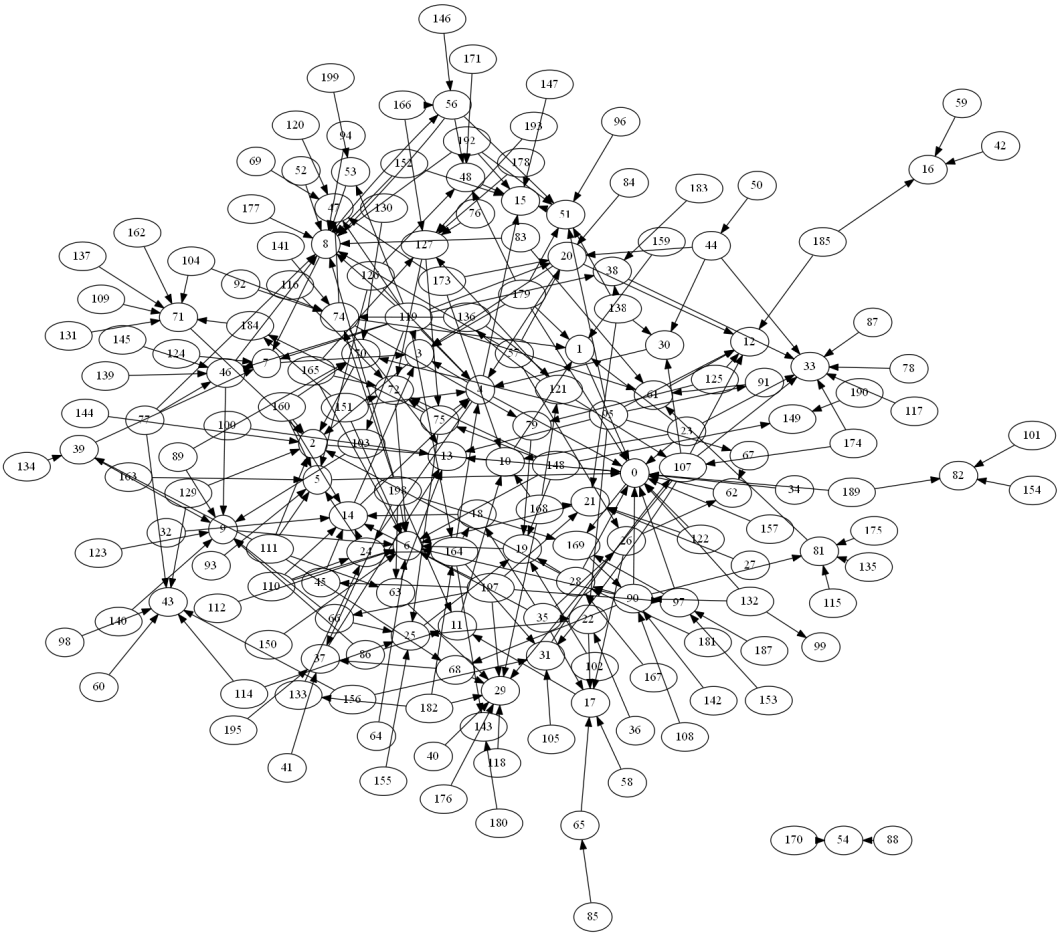
Vzorek			S1	S2	S3	S4	S5	S6	S7	S8	S9
vrcholy	hrany	fitness									
100	136	S1, S2	0.013	0.004	1.000	1.000	0.287	0.270	0.227	0.103	0.124
200	311	S1, S2	0.035	0.002	0.882	1.000	0.213	0.217	0.312	0.135	0.164
300	490	S1, S2	0.099	0.005	1.000	1.000	0.250	0.306	0.323	0.196	0.257
150	263	S1-S9	0.292	0.122	0.143	1.000	0.213	0.141	0.270	0.098	0.023

Tabulka 2: Výsledky redukce volební sítě Wikepedie



(a) Vstupní stupně (b) Výstupní stupně

Obrázek 12: Srovnání distribucí stupňů redukovaného grafu s původním grafem volební sítě



Obrázek 13: Vizualizace redukovaného grafu volební sítě Wikipedie

6.3 Redukce reálných sítí

6.3.1 Sociální síť Epinions

Na webu Epinions.com uživatelé publikují své recenze na nejrůznější produkty. Každý uživatel si může zvolit komu „věří“ a komu „nevěří“, na základě čehož se potom filtrují a setřizí recenze uživatelů. Tato relace tvoří orientovaný graf, kde hrana e_{ij} znamená, že uživatel i věří uživateli j [26]. Tato síť má distribuce vstupních i výstupních stupňů pocházející z mocninného rozdělení.

Genetický algoritmus jsem testoval celkem na 6 různých velikostí vzorku. Cílem bylo otestovat jak se algoritmus bude chovat pro různé velikosti vzorku a jak velký může být redukovaný vzorek, aby algoritmus byl schopen ještě pracovat, tedy aby na jedincích probíhala evoluce a nejlepší řešení postupně konvergovalo k vlastnostem původního grafu. Distribuce pro fitness funkci jsem opět zvolil S1 a S2.

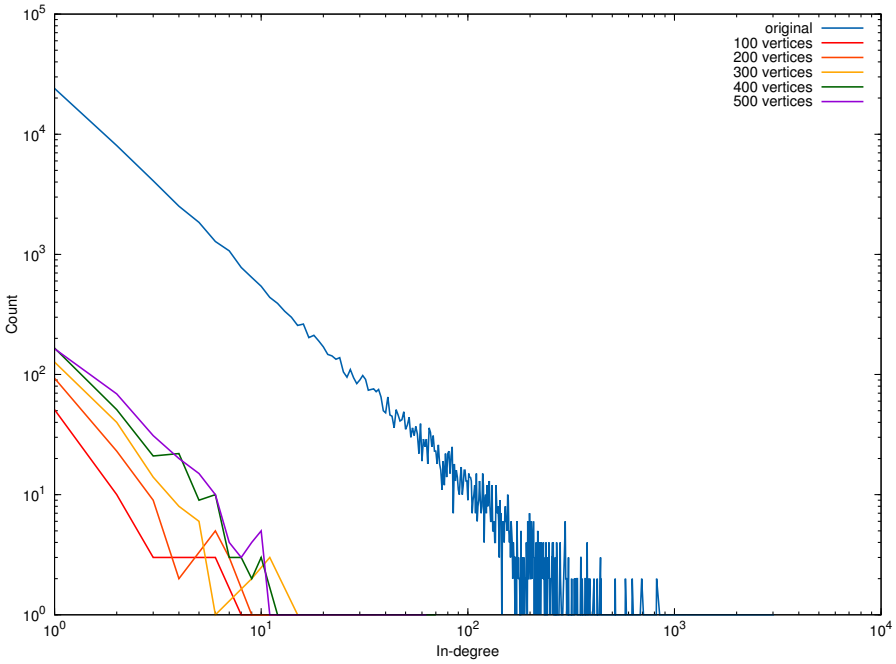
Pro první tři případy, kdy byly velikosti redukovaného vzorku 100, 200 a 300 uzlů algoritmus podával dobré výsledky. Odchylky od původních distribucí S1 a S2 nepřekročily 0,1. Pro nalezení takového řešení bylo zapotřebí maximálně 1000 generací. Výsledky jsou zobrazeny na obrázku 14 a shrnuty v tabulce 3.

Pro vzorky o velikosti 400 a 500 uzlů se konvergence genetického algoritmu už zpomalovala, redukované distribuce S1 a S2 mají ale stále nízké odchylky od distribucí původního grafu. Algoritmus běžel 1000 generací pro 500 vrcholů a 2000 generací pro velikost vzorku 400 vrcholů. Výsledky jsou zobrazeny na obrázku 14 a shrnuty v tabulce 3.

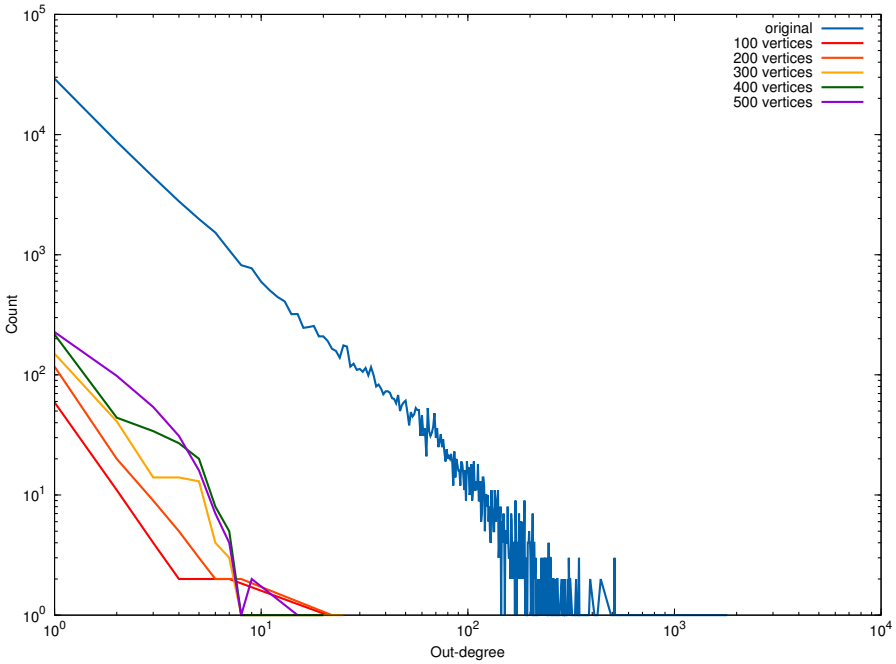
Dále byl testován vzorek o velikosti 600 uzlů, bohužel algoritmus v tomto případě nebyl schopen nalézt lepší řešení než vygenerované v počáteční generaci, tak jej ani neuvádím na obrázcích.

V posledním testu s touto sítí jsem adaptoval všech devět vlastností. Zvolil jsem velikost redukovaného vzorku na 150 vrcholů. Výsledky nejlepšího nalezeného řešení jsou shrnuty v tabulce 3 (řádek 6).

Maximální velikost redukovaného vzorku pro tuto síť lze dle výsledků testů stanovit na 500 vrcholů. Algoritmus nejlépe pracoval se vzorky o velikosti 300 až 400 vrcholů, kdy odchylky byly ještě poměrně nízké a rychlost konvergence vysoká. Celkově u všech grafů lze říct, že tvar distribuce S1 a S2 více méně zachovaly. V tabulce 3 lze pozorovat, že pro distribuci S3 žádný z redukovaných vzorků neodpovídá předloze, to je způsobeno tím, že původní distribuce má pouze dva prvky.



(a) Vstupní stupně



(b) Výstupní stupně

Obrázek 14: Srovnání distribucí stupňů pro různě velké redukované grafy sítě Epinions

Vzorek			S1	S2	S3	S4	S5	S6	S7	S8	S9
vrcholy	hrany	fitness									
100	136	S1, S2	0.006	0.005	1.000	1.000	0.356	0.377	0.237	0.561	0.450
200	283	S1, S2	0.017	0.022	1.000	0.014	0.186	0.210	0.317	0.509	0.291
300	483	S1, S2	0.009	0.070	1.000	0.015	0.193	0.222	0.319	0.561	0.554
400	735	S1, S2	0.059	0.152	1.000	0.019	0.162	0.164	0.352	0.488	0.474
500	900	S1, S2	0.076	0.254	1.000	0.014	0.193	0.213	0.359	0.516	0.354
150	299	S1-S9	0.196	0.334	1.000	0.012	0.096	0.188	0.237	0.144	0.037

Tabulka 3: Výsledky redukce sociální sítě Epinions

6.3.2 Citační síť

Arxiv HEP-PH (high energy physics phenomenology - publikace částicové fyziky) představuje citační graf z arXiv a pokrývá citace 34 546 publikací (vrcholy), které citují 421 578 jiných publikací (hrany). Orientovaná hrana e_{ij} znamená, že publikace i cituje publikace j. Pokud je publikace citována jinou publikací, která nepatří do této datové kolekce, tak informace o této citaci není v grafu zahrnuta. Data pocházejí z období od ledna 1993 do dubna 2003 [26].

Tuto síť jsem do testů zahrnul, protože jsem chtěl otestovat jak se algoritmus bude chovat pro ne zcela mocninné tvary distribucí S1 a S2. Síť jsem zkusil zredukovat pro sedm různých velikosti redukovaného grafu. Cílem bylo také otestovat jak si algoritmus poradí s různými velikosti vzorku. Distribuce pro fitness funkci jsem opět zvolil S1 a S2. Výsledky testu jsou zobrazeny na obrázku 15 a shrnuty v tabulce 4.

Pro tři nejmenší velikosti 100, 200 a 500 vrcholů algoritmus byl schopen řešení maximálně za 2 000 generací. Hodnoty odchylek pro S1 a S2 byly nízké $D \approx 0.01$.

Další testovaná velikost byla 800 vrcholů. Po 1 500 generacích bylo nejlepší nalezené řešení původně vytvořeno už v generaci 849 a odchylky pro S1 a S2 byly stále nízké. Pro tuto velikost redukovaného vzorku se rychlost konvergence velice zpomalila a výměna na pěti nejlepších řešeních byla občasná.

Pro 1 200 vrcholů byl výstupem graf s 3 543 hranami, ovšem pocházel už z 8. generace, algoritmus běžel ještě dalších cca 200 generací, ale evoluce už byla zastavena a dostala se do lokálního extrému. Pro velikost redukovaného vzorku 2 000 vrcholů se algoritmus choval podobně, nejlepší nalezený graf byl už z 5. generace. Zkusil jsem dále zvolit takovou maximální velikost vzorku, jakou mi paměť testovacího počítače dovolila, jednalo se o graf s 6 500 vrcholy. Výstupem byl graf vytvořen už ve 3. generaci a stále nízkými odchylkami, což je nejspíš způsobeno tím, že původní graf měl spíše náhodné rozdělení stupňů vrcholů.

Poslední experiment se týkal adaptace všech devíti vlastností grafu. Velikost redukovaného grafu byla 200 vrcholů. Výsledky jsou shrnuty v tabulce 4 (řádek 8). Vlastnosti redukovaného grafu se původním vlastnostem podobají s nízkými odchylkami, vyjma S8.

Z výsledků tohoto testu lze usuzovat, že maximální velikost redukovaného grafu, s jakou genetický algoritmus dokáže rozumně pracovat je do 1000 vrcholů, nejlépe evoluce

probíhala na grafech do velikosti 500 vrcholů. Tvary distribucí větších grafů se už příliš vzdalují původnímu grafu a lze pozorovat, že jejich rozdělení pochází spíše z Poissonova než z mocninného rozdělení.

Vzorek			S1	S2	S3	S4	S5	S6	S7	S8	S9
vrcholy	hrany	fitness									
100	255	S1, S2	0.009	0.008	1.000	1.000	0.116	0.123	0.208	0.579	0.246
200	600	S1, S2	0.011	0.003	1.000	1.000	0.078	0.090	0.275	0.589	0.192
500	1625	S1, S2	0.014	0.011	1.000	1.000	0.044	0.050	0.188	0.530	0.087
800	2347	S1, S2	0.029	0.010	1.000	0.004	0.047	0.068	0.174	0.438	0.269
1200	3543	S1, S2	0.043	0.020	1.000	1.000	0.076	0.045	0.080	0.287	0.265
2000	5903	S1, S2	0.032	0.024	1.000	1.000	0.056	0.051	0.111	0.186	0.133
6500	19723	S1, S2	0.085	0.088	1.000	1.000	0.138	0.113	0.126	0.295	0.181
200	632	S1-S9	0.074	0.034	0.200	0.065	0.108	0.081	0.162	0.441	0.052

Tabulka 4: Výsledky redukce citační sítě

6.3.3 Web Stanfordovy univerzity

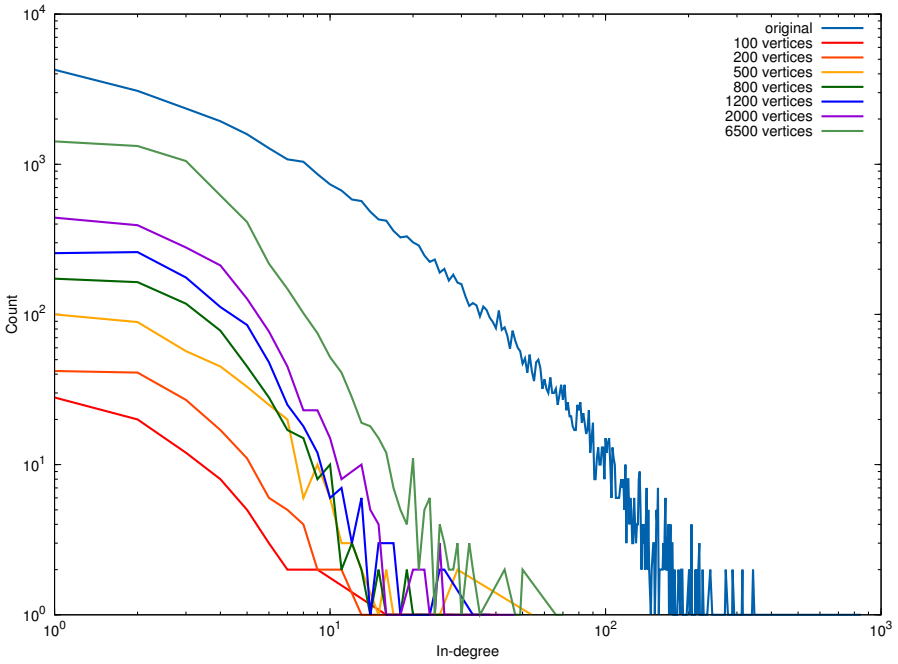
Univerzitní síť webových dokumentů byla sesbírána v roce 2002. Obsahuje 281 903 uzlů a 2 312 497 hran. Orientované hrany grafu reprezentují odkazy mezi jednotlivými dokumenty [26]. Tato síť má charakteristické distribuce slabě i silně souvislých komponent. Proto jedním z cílů bylo ověřit zda je genetický algoritmus chopen redukovaný graf upravit tak, aby i S3 a S4 distribuce napodobily charakteristický tvar těch původních.

První testovací vzorek měl velikost 300 uzlů, nejlepší řešení mělo 790 hran, genetický algoritmus jej našel po 200 generacích. Pro fitness hodnoty se počítaly odchylky S1 až S4 distribucí. Výsledky redukce je možno pozorovat na obrázku 16 a hodnoty odchylek jsou shrnuty v tabulce 5 (řádek 1). Lze pozorovat, že tvar distribuce vstupních a výstupních stupňů je více méně zachován, ale tvar distribucí slabě i silně souvislých komponent vůbec neodpovídá předloze.

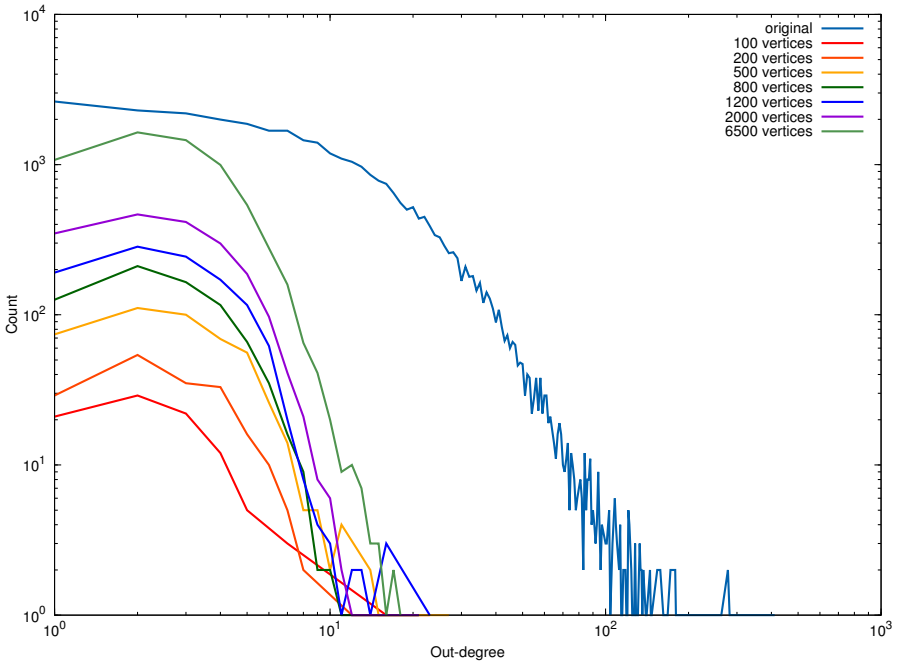
Neúspěch předchozího případu napodobit tvary S3 a S4 distribucí, mohl být způsoben malým redukovaným grafem. Další pokus tedy byl testován s velikostí redukovaných grafů 6500 vrcholů. Zkoušel jsem napodobit pouze S1 a S3 distribuce, algoritmus vrátil výsledek s 14 435 hranami, jehož distribuce jsou zobrazeny na obrázku 17 a odchylky shrnuty v tabulce 5 (řádek 2). Opět algoritmus nedokázal napodobit tvar distribuce S3, tvar S1 zůstal zachován.

Další experiment s touto sítí se týkal distribuce shlukovacího koeficientu S9. Na velikosti vzorku 500 vrcholů jsem chtěl ověřit, zda algoritmus dokáže upravit tvar distribuce S9 podle původní tak, aby byl zachován i S1 a S2. Výsledky jsou na obrázku 18 a shrnuty v tabulce 5 (řádek 3). Tvar S1 zůstal zachován i distribuce S9 redukovaného vzorku má poměrně nízkou odchylku od původní neredukované distribuce.

Úkolem posledního experimentu bylo ověřit případ, kdy bude adaptováno všech devět parametrů redukovaného grafu. Tedy pro výpočet fitness funkce byly zahrnuty distribuce



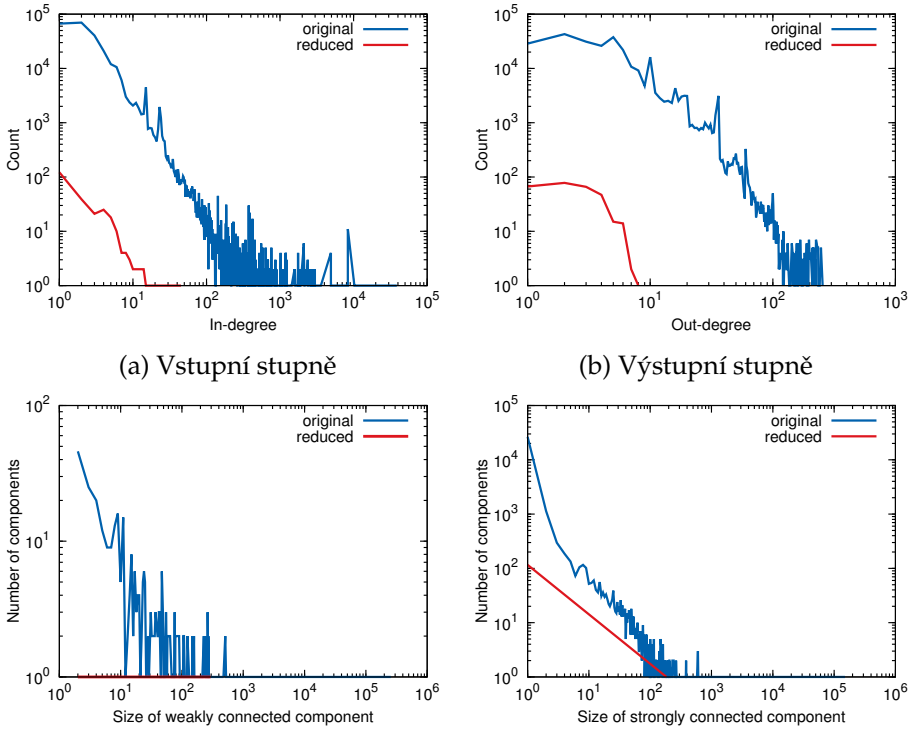
(a) Vstupní stupně



(b) Výstupní stupně

Obrázek 15: Srovnání distribucí stupňů pro různě velké redukované grafy citační sítě

S1 až S9. Genetický algoritmus jsem spouštěl opakovaně, čím větší byla velikost redukovaného grafu, tím větší měl algoritmus tendenci zasekávat se v lokálních extrémech. Další výsledky jsou v tabulce 5 (řádek 4-5). Společným rysem výsledných grafů byla vysoká odchylka u distribuce vstupních stupňů, což byla jediná mocninná distribuce původního grafu. Ostatní vlastnosti byly zpravidla zachovány většinou poměrně dobře s nízkými odchylkami od tvaru původní distribuce. Pro redukovaný graf o velikosti 150 vrcholů jsou hodnoty odchylek v tabulce 5 (řádek 4) a distribuce jsou porovnány na obrázku 19 (distribuce S8 u redukovaného grafu je natažená v ose x tak, aby byl vidět její tvar.)



(a) Vstupní stupně

(b) Výstupní stupně

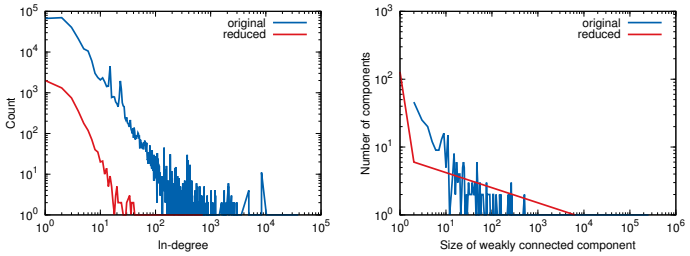
(c) Slabě souvislé komponenty

(d) Silně souvislé komponenty

Obrázek 16: Srovnání distribucí redukovaného a původního grafu

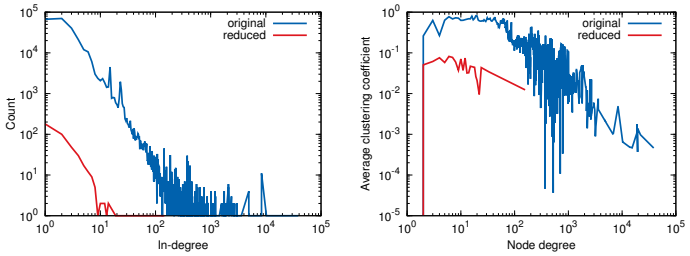
Vzorek			S1	S2	S3	S4	S5	S6	S7	S8	S9
vrcholy	hrany	fitness									
300	790	S1-S4	0.278	0.345	0.084	0.791	0.153	0.243	0.146	0.573	0.411
6500	14435	S1, S3	0.162	0.118	0.648	1.000	0.090	0.059	0.194	0.595	0.169
500	1206	S1, S9	0.168	0.172	1.000	1.000	0.103	0.103	0.407	0.691	0.023
150	357	S1-S9	0.407	0.059	0.160	0.006	0.158	0.166	0.100	0.230	0.103
200	584	S1-S9	0.398	0.096	0.333	0.008	0.164	0.162	0.088	0.345	0.125

Tabulka 5: Výsledky redukce sítě webových dokumentů ze Stanfordovy univerzity



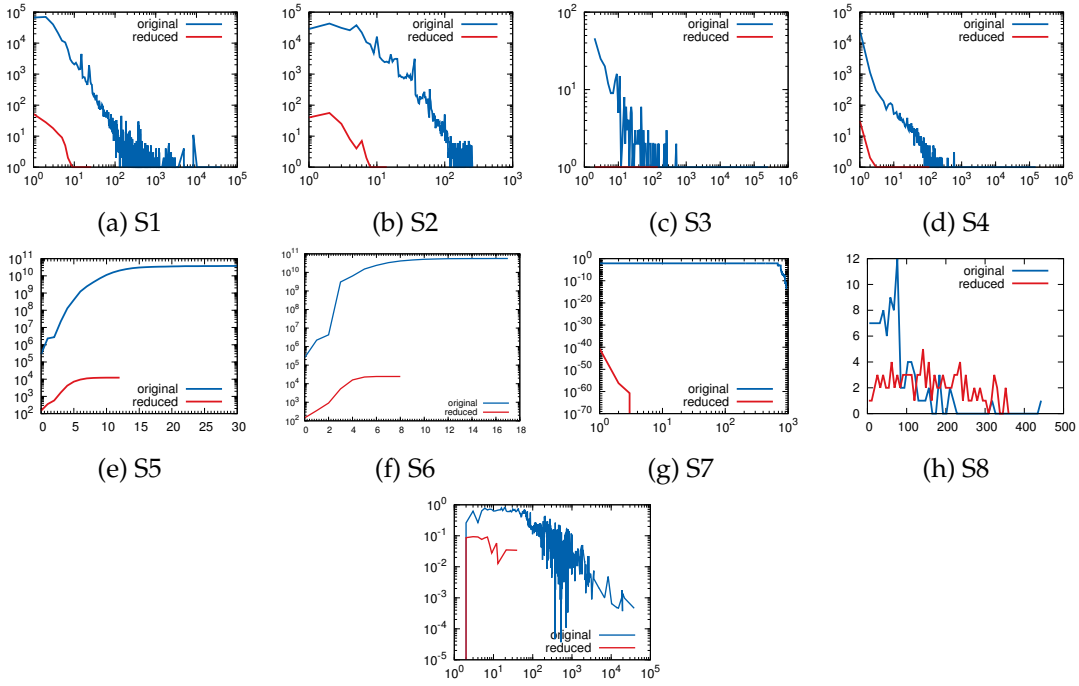
(a) Vstupní stupeň (b) Slabě souvislé komp.

Obrázek 17: Srovnání slabě souvislých komponent a vstupních stupňů pro web



(a) Vstupní stupně (b) Shlukovací koeficient

Obrázek 18: Srovnání shlukovacího koeficientu



(a) S1 (b) S2 (c) S3 (d) S4

(e) S5 (f) S6 (g) S7 (h) S8

(i) S9

Obrázek 19: Srovnání redukovaných a původních distribucí pro univerzitní web

7 Závěr

Tato diplomová práce přináší přehled vlastností charakterizujících bezškálové sítě, zmiňuje se o stávajících redukčních metodách a představuje základní informace o genetických algoritmech. Hlavní náplní ovšem je návrh a implementace genetického algoritmu pro redukci bezškálových sítí.

Vzniklou aplikaci jsem otestoval celkem na čtyřech reálných sítích. Výsledky testů napovídají, že algoritmus dokáže uspokojivě pracovat pouze s malými redukovanými grafy. Maximální velikost, kdy na jedincích ještě probíhá evoluce a vlastnosti nejlepšího řešení konvergují k vlastnostem původního neredukovaného grafu, je závislá na vlastnostech původní sítě, volbě parametrů genetického algoritmu a také na volbě vlastností pro fitness funkci. Při adaptaci pouze distribucí vrcholových stupňů byla ve většině případů největší možná velikost redukovaných grafů 500 vrcholů. V případech adaptace všech devíti vlastností byla velikost redukovaných grafů, se kterou algoritmus dobře pracoval, okolo 200 vrcholů. U takto malých grafů byly průměrné hodnoty odchylek distribucí od distribucí původního grafu přibližně $0,2$, což je výsledek srovnatelný se současnými redukčními metodami založenými na algoritmu náhodné procházky grafem. Nicméně je nutné brát v potaz rozdílnou velikost redukovaných grafů. Z výsledků experimentů lze také říci, že vliv na míru adaptace vlastností redukovaného grafu neměla ani tak velikost původní sítě, jako spíše tvar distribucí původní sítě. Z výše zmíněných faktů lze usuzovat, že vhodnou aplikací tohoto genetického algoritmu je například vizualizace sítí.

Implementovaný genetický algoritmus by se v budoucnu dal rozvinout několika způsoby. Jeden z nich by mohlo být zvolení odlišné reprezentace jedince tak, že by netvořil kompletně nový graf, ale obsahoval by pouze seznam odkazů na vrcholy původního neredukovaného grafu. Výhoda tohoto přístupu by mohla být v tom, že bude možné určit, které vrcholy z původního grafu jsou obsaženy také v redukovaném grafu. Další možné vylepšení se týká fitness funkce, která by mohla být realizována tak, že by porovnávala sklony distribuce stupňů vrcholů a minimální hodnoty stupňů odkud se začínají projevovat bezškálové vlastnosti redukovaného a původního grafu.

Martin Střílka

8 Reference

[1] BARABÁSI, Albert-László. *V pavučině sítí*. Vyd. 1. V Praze: Paseka, 2005, 280 s. ISBN 80-718-5751-3. [cit. 2014-04-30].

[2] AFFENZELLER, Michael. *Genetic algorithms and genetic programming: modern concepts and practical applications*. Boca Raton: CRC Press, c2009, xxvii, 365 s. ISBN 978-1-58488-629-7.

[3] LESKOVEC, Jure a Christos FALOUTSOS. *Sampling from Large Graphs*. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2006.

[4] GILBERT, A. C. a K. LEVCHENKO. *Compressing network graphs*. LinkKDD, 2004.

[5] LESKOVEC, Jure, Jon KLEINBERG a Christopher FALOUTSOS. *Graph Evolution: Densification and Shrinking Diameters*. ACM Transactions on Knowledge Discovery from Data, 2007.

[6] KUMAR, R.,BRODER, A., MAGHOUL, F., RAGHAVAN, P., RAJAGOPALAN, S., STATA, R., TOMKINS, A., ANDWIENER, J. *Graph structure in the web: experiments and models*. Proceedings of World Wide Web Conference, 2000.

[7] POŠÍK, Petr. *Paralelní genetické algoritmy*. Praha, 2001. Diplomová práce. České vysoké učení technické.

[8] HOLLAND, John H. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. Cambridge: Bradford Book, c1992, xiv, 211 s. Complex adaptive systems. ISBN 02-625-8111-6.

[9] KUREKOVÁ, Iveta. *Určenie modelu komplexnej siete*. Ostrava, 2013. Bakalářská práce. VŠB – Technická univerzita Ostrava.

[10] MILGRAM, Stanley. *The small world problem*. New York: Harper and Row, 1969.

[11] *Network Science* [online]. [cit. 2014-04-30]. Dostupné z: `http://barabasilab.neu.edu/networksciencebook/downlPDF.html`

[12] GRANOVETTER, Mark. The Strength of Weak Ties. *American Journal of Sociology*. 1973. Dostupné z: `http://sociology.stanford.edu/people/mgranovetter/documents/granstrengthweakties.pdf`

[13] WATTS, D. J. a S. H. STROGATZ. Collective dynamics of 'small-world' networks. *Nature*. 1998.

[14] *Grafové algoritmy a komplexní sítě* [online]. [cit. 2014-04-30]. Dostupné z: `http://www.cs.vsb.cz/ochodkova/`

[15] GLADWELL, Malcolm. *The tipping point: how little things can make a big difference*. 1st Back Bay pbk. ed. Boston: Back Bay Books, 2002, xii, 301 s. ISBN 978-0-316-34662-7.

[16] *The Oracle of Bacon* [online]. [cit. 2014-04-30]. Dostupné z: http://oracleofbacon.org/

[17] NEWMAN, M. *Networks: an introduction*. New York: Oxford University Press, 2010, xi, 772 s. ISBN 978-0-19-920665-0.

[18] ALBERT, Réka a Albert-László BARABÁSI. *Statistical mechanics of complex networks*. 2002. Dostupné z: http://www.barabasilab.com/pubs/CCNR-ALB_Publications/200201-30_RevModernPhys-StatisticalMech/200201-30_RevModernPhys-StatisticalMech.pdf

[19] Connected Components. [online]. [cit. 2014-05-01]. Dostupné z: https://www8.cs.umu.se/kurser/TDBA77/VT06/algorithms/BOOK/BOOK4/NODE159.HTM

[20] Hop-Plot. [online]. [cit. 2014-05-01]. Dostupné z: http://konect.uni-koblenz.de/plots/hop_plot

[21] DARWIN, Charles. *O původu druhů prostřednictvím přírodního výběru aneb Záchrana preferovaných ras v existenčním boji*. 1859.

[22] HROMEK, Martin. *Genetické algoritmy a MPI*. Diplomová práce. VŠB - Technická univerzita Ostrava.

[23] *Evoluční výpočetní techniky: principy a aplikace*. 1. české vyd. Praha: BEN, 2009, 534 s. ISBN 978-80-7300-218-3.

[24] GEORGY, Maged a Sameh Y. BASILY. Using genetic algorithms in optimizing construction material delivery schedules. *Construction Innovation: Information, Process, Management*. 2008, vol. 8, issue 1, s. 23-45. DOI: 10.1108/14714170810846503. Dostupné z: http://www.emeraldinsight.com/10.1108/14714170810846503

[25] *Stanford Network Analysis Platform* [online]. [cit. 2014-05-01]. Dostupné z: http://snap.stanford.edu/

[26] Stanford Large Network Dataset Collection. [online]. [cit. 2014-05-01]. Dostupné z: http://snap.stanford.edu/data/